



USPTO

[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

 Search: ☒ The ACM Digital Library ☐ The Guide

+critical +code +optimiz\*


[Feedback](#) [Report a problem](#) [Satisfaction survey](#)
Terms used: **critical code optimiz**

Found 11,886 of 215,737

Sort results by

relevance

Display results

expanded form

[Save results to a Binder](#)[Search Tips](#)☐ Open results in a new window[Try an Advanced Search](#)[Try this search in The ACM Guide](#)

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance scale ☐ ☐ ☐ ☐ ☐

### 1 [Code optimization - I: Compiler optimization-space exploration](#)

Spyridon Triantafyllis, Manish Vachharajani, Neil Vachharajani, David I. August  
 March 2003 **Proceedings of the international symposium on Code generation and optimization: feedback-directed and runtime optimization CGO '03**

Publisher: IEEE Computer Society

Full text available: [pdf\(1.19 MB\)](#)
 Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

To meet the demands of modern architectures, optimizing compilers must incorporate an ever larger number of increasingly complex transformation algorithms. Since code transformations may often degrade performance or interfere with subsequent transformations, compilers employ predictive heuristics to guide optimizations by predicting their effects a priori. Unfortunately, the unpredictability of optimization interaction and the irregularity of today's wide-issue machines severely limit the accuracy ...

### 2 [Low Power Embedded Software Optimization Using Symbolic Algebra](#)

A. Peymandoust, T. Simunic, G. de Micheli  
 March 2002 **Proceedings of the conference on Design, automation and test in Europe DATE '02**

Publisher: IEEE Computer Society

Full text available: [pdf\(119.80 KB\)](#) Additional Information: [full citation](#), [abstract](#), [citations](#)

The market demand for portable multimedia applications has exploded in the recent years. Unfortunately, for such applications current compilers and software optimization methods often require designers to do part of the optimization manually. Specifically, the high-level arithmetic optimizations and the use of complex instructions are left to the designers' ingenuity. In this paper, we present a tool flow, SymSoft, that automates the optimization of power-intensive algorithmic constructs using symbolic algebra ...

### 3 [High-level power estimation \(invited talks\): Source code optimization and profiling of energy consumption in embedded systems](#)

Tajana Šimunić, Luca Benini, Giovanni De Micheli, Mat Hans  
 September 2000 **Proceedings of the 13th international symposium on System synthesis ISSS '00**

Publisher: IEEE Computer Society

Full text available: [pdf\(81.88 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

This paper presents a source code optimization methodology and a profiling tool that have been developed to help designers in optimizing software performance and energy in embedded systems. Code optimizations are applied at three levels of abstraction: algorithmic, data and instruction-level. The profiler exploits a cycle-accurate energy consumption simulator [3] to relate the embedded system energy consumption and

performance to the source code. Thus, it can be used for analysis (i.e., to find ...

4 Code compression: Compiler optimization and ordering effects on VLIW code compression



Montserrat Ros, Peter Sutton

October 2003 **Proceedings of the 2003 international conference on Compilers, architecture and synthesis for embedded systems CASES '03**

Publisher: ACM Press

Full text available: pdf(334.18 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Code size has always been an important issue for all embedded applications as well as larger systems. Code compression techniques have been devised as a way of battling bloated code; however, the impact of VLIW compiler methods and outputs on these compression schemes has not been thoroughly investigated. This paper describes the application of single- and multiple-instruction dictionary methods for code compression to decrease overall code size for the TI TMS320C6xxx DSP family. The compression ...

**Keywords:** VLIW, code compression, compiler optimizations

5 Optimizing for reduced code space using genetic algorithms



Keith D. Cooper, Philip J. Schielke, Devika Subramanian

May 1999 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1999 workshop on Languages, compilers, and tools for embedded systems LCTES '99**, Volume 34 Issue 7

Publisher: ACM Press

Full text available: pdf(977.31 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Code space is a critical issue facing designers of software for embedded systems. Many traditional compiler optimizations are designed to reduce the execution time of compiled code, but not necessarily the size of the compiled code. Further, different results can be achieved by running some optimizations more than once and changing the order in which optimizations are applied. Register allocation only complicates matters, as the interactions between different optimizations can cause more spill c ...

6 Compiler analysis and optimization: General loop fusion technique for nested loops considering timing and code size



Meilin Liu, Qingfeng Zhuge, Zili Shao, Edwin H.-M. Sha

September 2004 **Proceedings of the 2004 international conference on Compilers, architecture, and synthesis for embedded systems CASES '04**

Publisher: ACM Press

Full text available: pdf(307.40 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Loop fusion is commonly used to improve the instruction-level parallelism of loops for high-performance embedded computing systems. Loop fusion, however, is not always directly applicable because the fusion prevention dependencies may exist among loops. Most of the existing techniques still have limitations in fully exploiting the advantages of loop fusion. In this paper, we present a general loop fusion technique for loops or nested loops based on the loop dependency graph model, retiming, and ...

**Keywords:** code size, embedded DSP, loop fusion, retiming, scheduling

7 Optimizing Address Code Generation for Array-Intensive DSP Applications

Guilin Chen, Mahmut Kandemir

March 2005 **Proceedings of the international symposium on Code generation and optimization CGO '05**

Publisher: IEEE Computer Society

Full text available: pdf(523.04 KB) Additional Information: [full citation](#), [abstract](#), [index terms](#)

The application code size is a critical design factor for many embedded systems. Unfortunately, most available compilers optimize primarily for speed of execution rather than code density. As a result, the compiler-generated code can be much larger than necessary. In particular, in the DSP domain, the past research found that optimizing address code generation can be very important since address code can account for over 50% of all program bits. This paper presents a compiler-directed scheme to ...

## 8 Ada software engineering and optimized code



Gary Frankel

July 1989 **Proceedings of the conference on TRI-Ada '88 TRI-Ada '88**

**Publisher:** ACM Press

Full text available: [pdf\(518.89 KB\)](#)

Additional Information: [full citation](#), [index terms](#)

## 9 Pipelined Execution of Critical Sections Using Software-Controlled Caching in Network Processors



Jinquan Dai, Long Li, Bo Huang

March 2007 **Proceedings of the International Symposium on Code Generation and Optimization CGO '07**

**Publisher:** IEEE Computer Society

Full text available: [pdf\(9.78 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [index terms](#)

To keep up with the explosive internet packet processing demands, modern network processors (NPs) employ a highly parallel, multi-threaded and multi-core architecture. In such a parallel paradigm, accesses to the shared variables in the external memory (and the associated memory latency) are contained in the critical sections, so that they can be executed atomically and sequentially by different threads in the network processor. In this paper, we present a novel program transformation that is us ...

## 10 Critical path reduction for scalar programs



Michael Schlansker, Vinod Kathail

December 1995 **Proceedings of the 28th annual international symposium on Microarchitecture MICRO 28**

**Publisher:** IEEE Computer Society Press

Full text available: [pdf\(1.38 MB\)](#)

Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

## 11 Short presentations with posters: SCCP/x: a compilation profile to support testing and verification of optimized code



Raimund Kirner

September 2007 **Proceedings of the 2007 international conference on Compilers, architecture, and synthesis for embedded systems CASES '07**

**Publisher:** ACM Press

Full text available: [pdf\(350.81 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Embedded systems are often used in safety-critical environments. Thus, thorough testing of them is mandatory. A quite active research area is the automatic test-case generation for testing embedded systems. To achieve high retargetability of the testing framework, the test-case generation has to be done at source-code level. However, it is challenging to guarantee that the test-cases obtained from the source code are also valid at the object-code level, since even in safety-critical domains pr ...

**Keywords:** code transformation, compiler, decision coverage, optimization, structural code-coverage preservation, testing

## 12 Dynamic translation: Metadata driven memory optimizations in dynamic binary



translator

Chaohao Xu, Jianhui Li, Tao Bao, Yun Wang, Bo Huang

June 2007 **Proceedings of the 3rd international conference on Virtual execution environments VEE '07****Publisher:** ACM PressFull text available: [pdf\(3.42 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

A dynamic binary translator offers solutions for translating and running source architecture binaries on target architecture at runtime. Regardless of its growing popularity, practical dynamic binary translators usually suffer from the limited optimizations performed when generating the translated code due to the lack of useful information available in the executable files and the requirement to conform to the binary-level compatibility. Trying to generate more efficient translated code, we p ...

**Keywords:** dynamic binary translator, memory optimizations, metadata

13 A study of source-level compiler algorithms for automatic construction of pre-execution code



Dongkeun Kim, Donald Yeung

August 2004 **ACM Transactions on Computer Systems (TOCS)**, Volume 22 Issue 3**Publisher:** ACM PressFull text available: [pdf\(1.55 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Pre-execution is a promising latency tolerance technique that uses one or more helper threads running in spare hardware contexts ahead of the main computation to trigger long-latency memory operations early, hence absorbing their latency on behalf of the main computation. This article investigates several source-to-source C compilers for extracting pre-execution thread code automatically, thus relieving the programmer or hardware from this onerous task. We present an aggressive profile-driven co ...

**Keywords:** Data prefetching, memory-level parallelism, multithreading, pre-execution, prefetch conversion, program slicing, speculative loop parallelization

14 Optimized code restructuring of OS/2 executables

Jyh-Herng Chow, Yong-fong Lee, Kalyan Muthukumar, Vivek Sarkar, Mauricio Serrano, Iris Garcia, John Hsu, Shauchi Ong, Honesty Young

November 1995 **Proceedings of the 1995 conference of the Centre for Advanced Studies on Collaborative research CASCON '95****Publisher:** IBM PressFull text available: [pdf\(234.83 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

This paper describes the design and algorithms of FDPR/2 (Feedback Directed Program Restructuring of OS/2 executables), a general-purpose tool that can be used to instrument, profile, and restructure/optimize OS/2 executables for the tel x86 architecture. The optimizations delivered by FDPR/2's restructuring include improved utilization of the (instruction) memory hierarchy, improved branch alignment, and dead code elimination. These optimizations are known to be critical for object-oriented pro ...

15 Improving WCET by applying a WC code-positioning optimization



Wankang Zhao, David Whalley, Christopher Healy, Frank Mueller

December 2005 **ACM Transactions on Architecture and Code Optimization (TACO)**, Volume 2 Issue 4**Publisher:** ACM PressFull text available: [pdf\(510.31 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Applications in embedded systems often need to meet specified timing constraints. It is advantageous to not only calculate the worst-case execution time (WCET) of an

application, but to also perform transformation, which reduce the WCET, since an application with a lower WCET will be less likely to violate its timing constraints. Some processors incur a pipeline delay whenever an instruction transfers control to a target that is not the next sequential instruction. Code-positioning optimizations ...

**Keywords:** WCET, code positioning, embedded systems

16 Rapid development of optimized DSP code from a high level description through software estimations



Alain Pegatoquet, Emmanuel Gresset, Michel Auguin, Luc Bianco

June 1999 **Proceedings of the 36th ACM/IEEE conference on Design automation DAC '99**

**Publisher:** ACM Press

Full text available: pdf(51.51 KB) Additional Information: [full citation](#), [references](#), [index terms](#)

**Keywords:** DSP, code generation, performance estimation

17 Architectural design for embedded systems: Design space minimization with timing and code size optimization for embedded DSP



Qingfeng Zhuge, Zili Shao, Bin Xiao, Edwin H.-M. Sha

October 2003 **Proceedings of the 1st IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis CODES+ISSS '03**

**Publisher:** ACM Press

Full text available: pdf(131.73 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

One of the most challenging problems in high-level synthesis is how to quickly explore a wide range of design options to achieve high-quality designs. This paper presents an Integrated Framework for Design Optimization and Space Minimization (IDOM) towards finding the minimum configuration satisfying timing and code size constraints. We show an effective way to reduce the design space to be explored through the study of the fundamental properties and relations among multiple design parameters, s ...

**Keywords:** DSP processors, code size reduction, retiming, unfolding

18 Source-level global optimizations for fine-grain distributed shared memory systems



R. Veldema, R. F. H. Hofman, R. A. F. Bhoedjang, C. J. H. Jacobs, H. E. Bal

June 2001 **ACM SIGPLAN Notices , Proceedings of the eighth ACM SIGPLAN symposium on Principles and practices of parallel programming PPoPP '01**, Volume 36 Issue 7

**Publisher:** ACM Press

Full text available: pdf(112.60 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper describes and evaluates the use of aggressive static analysis in Jackal, a fine-grain Distributed Shared Memory (DSM) system for Java. Jackal uses an optimizing, source-level compiler rather than the binary rewriting techniques employed by most other fine-grain DSM systems. Source-level analysis makes existing access-check optimizations (e.g., access-check batching) more effective and enables two novel fine-grain DSM optimizations: object-graph aggregatio ...

19 Interaction cost and shotgun profiling



Brian A. Fields, Rastislav Bodik, Mark D. Hill, Chris J. Newburn

September 2004 **ACM Transactions on Architecture and Code Optimization (TACO)**, Volume 1 Issue 3

**Publisher:** ACM Press

Additional Information:

Full text available:  pdf(647.17 KB)[full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We observe that the challenges software optimizers and microarchitects face every day boil down to a single problem: bottleneck analysis. A bottleneck is any event or resource that contributes to execution time, such as a critical cache miss or window stall. Tasks such as tuning processors for energy efficiency and finding the right loads to prefetch all require measuring the performance costs of bottlenecks. In the past, simple event counts were enough to find the important bottlenecks. Today, t ...

**Keywords:** Performance analysis, critical path, modeling, profiling


## 20 [Partial dead code elimination using slicing transformations](#)



Rastislav Bodík, Rajiv Gupta

May 1997 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1997 conference on Programming language design and implementation PLDI '97**, Volume 32  
Issue 5

Publisher: ACM Press

Full text available:  pdf(1.80 MB)[Additional Information: full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We present an approach for optimizing programs that uncovers additional opportunities for optimization of a statement by *predicating* the statement. In this paper predication algorithms for achieving partial dead code elimination (PDE) are presented. The process of predication embeds a statement in a control flow structure such that the statement is executed only if the execution follows a path along which the value computed by the statement is live. The control flow restructuring performs ...

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)



USPTO

[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

 Search: ☒ The ACM Digital Library ☐ The Guide

+nest\* +loop +optimiz\*



THE ACM DIGITAL LIBRARY


[Feedback](#) [Report a problem](#) [Satisfaction survey](#)

Terms used: nest loop optimiz

Found 4,924 of 215,737

Sort results  
by

relevance

Display  
results

expanded form

[Save results to a Binder](#)[Search Tips](#)☐ Open results in a new window[Try an Advanced Search](#)[Try this search in The ACM Guide](#)

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance scale ☐ ☐ ☐ ☐ ☐1 [A compiler algorithm for optimizing locality in loop nests](#)

M. Kandemir, J. Ramanujam, A. Choudhary

July 1997 **Proceedings of the 11th international conference on Supercomputing ICS '97**

Publisher: ACM Press

Full text available: [pdf\(1.08 MB\)](#)Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)2 [A hyperplane based approach for optimizing spatial locality in loop nests](#)

M. Kandemir, A. Choudhary, N. Shenoy, P. Banerjee, J. Ramanujam

July 1998 **Proceedings of the 12th international conference on Supercomputing ICS '98**

Publisher: ACM Press

Full text available: [pdf\(1.13 MB\)](#)Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)3 [Optimized unrolling of nested loops](#)

Vivek Sarkar

May 2000 **Proceedings of the 14th international conference on Supercomputing ICS '00**

Publisher: ACM Press

Full text available: [pdf\(1.10 MB\)](#)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

In this paper, we address the problems of automatically selecting unroll factors for perfectly nested loops, and generating compact code for the selected unroll factors. Compared to past work, the contributions of our work include a) a more detailed cost model that includes ILP and 1-cache considerations, b) a new code generation algorithm for unrolling nested loops that generates more compact code (with fewer remainder loops) than the unroll-and-jam transf ...

4 [A compiler optimization to reduce execution time of loop nest](#)

Oh-Young Kwon, Gi-Ho Park, Tack-Don Han

March 1996 **ACM SIGARCH Computer Architecture News**, Volume 24 Issue 1

Publisher: ACM Press

Full text available: [pdf\(333.41 KB\)](#)Additional Information: [full citation](#), [abstract](#), [index terms](#)

In this paper, a compiler optimization to reduce the execution time of loop nest is proposed. Loop tiling is used to optimize loop nest. Loop tiling is the well-known optimization for improving locality. However, it has a count result that increases the

number of instructions to loop control. These increased instructions disturb the effect of locality optimization. Therefore, the tiling of innermost loop is not performed in order to reduce the instructions for loop control in this paper. This opti ...

##### 5 Automatic memory layout transformations to optimize spatial locality in parameterized loop nests



Philippe Clauss, Benoît Meister

March 2000 **ACM SIGARCH Computer Architecture News**, Volume 28 Issue 1

**Publisher:** ACM Press

Full text available: [pdf\(537.80 KB\)](#) Additional Information: [full citation](#), [abstract](#), [index terms](#)

One of the most efficient ways to improve program performances onto nowadays computers is to optimize the way cache memories are used. In particular, many scientific applications contain loop nests that operate on large multi-dimensional arrays whose sizes are often parameterized. No special attention is paid to cache memory performance when such loops are written. In this work, we focus on spatial locality optimization such that all the data that are loaded as a block in the cache will be used ...

**Keywords:** Ehrhart polynomials, cache memory, loop nests, optimizing compiler, parameterized polyhedron, program performance optimization, spatial locality

##### 6 Session S4.2: program transformation: Optimizing inter-nest data locality



M. Kandemir, I. Kadayif, A. Choudhary, J. A. Zambreno

October 2002 **Proceedings of the 2002 international conference on Compilers, architecture, and synthesis for embedded systems CASES '02**

**Publisher:** ACM Press

Full text available: [pdf\(272.47 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

By examining data reuse patterns of four array-intensive embedded applications, we found that these codes exhibit a significant amount of inter-nest reuse (i. e., the data reuse that occurs between different nests). While traditional compiler techniques that target array-intensive applications can exploit intra-nest data reuse, there has not been much success in the past in taking advantage of inter-nest data reuse. In this paper, we present a compiler strategy that optimizes inter-nest reuse ...

**Keywords:** array-intensive codes, cache locality, data reuse, embedded applications, inter-nest optimization

##### 7 Quantifying loop nest locality using SPEC'95 and the perfect benchmarks



Kathryn S. McKinley, Olivier Temam

November 1999 **ACM Transactions on Computer Systems (TOCS)**, Volume 17 Issue 4

**Publisher:** ACM Press

Full text available: [pdf\(635.63 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This article analyzes and quantifies the locality characteristics of numerical loop nests in order to suggest future directions for architecture and software cache optimizations. Since most programs spend the majority of their time in nests, the vast majority of cache optimization techniques target loop nests. In contrast, the locality characteristics that drive these optimizations are usually collected across the entire application rather than at the nest level. Researchers have studied nu ...


##### 8 A quantitative analysis of loop nest locality



Kathryn S. McKinley, Olivier Temam

September 1996 **ACM SIGPLAN Notices , ACM SIGOPS Operating Systems Review , Proceedings of the seventh international conference on Architectural support for programming languages and operating systems ASPLOS-VII**, Volume 31 , 30 Issue 9 , 5



**Publisher:** ACM PressFull text available:  [pdf\(1.49 MB\)](#)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper analyzes and quantifies the locality characteristics of numerical loop nests in order to suggest future directions for architecture and software cache optimizations. Since most programs spend the majority of their time in nests, the vast majority of cache optimization techniques target loop nests. In contrast, the locality characteristics that drive these optimizations are usually collected across the entire application rather than the nest level. Indeed, researchers have studied nume ...

## 9 [Control Flow Driven Splitting of Loop Nests at the Source Code Level](#)

Heiko Falk, Peter Marwedel

March 2003 **Proceedings of the conference on Design, Automation and Test in Europe - Volume 1 DATE '03****Publisher:** IEEE Computer SocietyFull text available:  [pdf\(215.57 KB\)](#)Additional Information: [full citation](#), [abstract](#), [index terms](#)[Publisher Site](#)

This paper presents a novel source code transformation for control flow optimization called loop nest splitting which minimizes the number of executed if-statements in loop nests of embedded multimedia applications. The goal of the optimization is to reduce runtimes and energy consumption. The analysis techniques are based on precise mathematical models combined with genetic algorithms. Due to the inherent portability of source code transformations, a very detailed benchmarking using 10 differen ...

## 10 [Integrating loop and data optimizations for locality within a constraint network based framework](#)

Guilin Chen, O. Ozturk, M. Kandemir, I. Kolcu


May 2005 **Proceedings of the 2005 IEEE/ACM International conference on Computer-aided design ICCAD '05****Publisher:** IEEE Computer SocietyFull text available:  [pdf\(234.62 KB\)](#)Additional Information: [full citation](#), [abstract](#), [citations](#)

In the context of data-intensive embedded applications, there have been two complementary approaches to data locality problem: restructuring code and restructuring data layout. Conceivably, an integrated approach that combines these two can generate much better results than each individual approach. However, there is an inherent difficulty in optimizing both data layout and loop access pattern simultaneously under a unified setting. This difficulty occurs due to the fact that a given data struct ...

## 11 [Optimization of array accesses by collective loop transformations](#)



Vivek Sarkar, Guang R. Gao

June 1991 **Proceedings of the 5th international conference on Supercomputing ICS '91****Publisher:** ACM PressFull text available:  [pdf\(1.10 MB\)](#)Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

## 12 [Blocking and array contraction across arbitrarily nested loops using affine partitioning](#)



Amy W. Lim, Shih-Wei Liao, Monica S. Lam

June 2001 **ACM SIGPLAN Notices , Proceedings of the eighth ACM SIGPLAN symposium on Principles and practices of parallel programming PPoPP '01**, Volume 36 Issue 7**Publisher:** ACM PressFull text available:  [pdf\(290.60 KB\)](#)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Applicable to arbitrary sequences and nests of loops, affine partitioning is a program transformation framework that unifies many previously proposed loop transformations,

including unimodular transforms, fusion, fission, reindexing, scaling and statement reordering. Algorithms based on affine partitioning have been shown to be effective for parallelization and communication minimization. This paper presents algorithms that improve data locality using affine partitioning. Blockin ...

13 Compiler analysis and optimization: General loop fusion technique for nested loops considering timing and code size



Meilin Liu, Qingfeng Zhuge, Zili Shao, Edwin H.-M. Sha

September 2004 **Proceedings of the 2004 international conference on Compilers, architecture, and synthesis for embedded systems CASES '04**

Publisher: ACM Press

Full text available: pdf(307.40 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Loop fusion is commonly used to improve the instruction-level parallelism of loops for high-performance embedded computing systems. Loop fusion, however, is not always directly applicable because the fusion prevention dependencies may exist among loops. Most of the existing techniques still have limitations in fully exploiting the advantages of loop fusion. In this paper, we present a general loop fusion technique for loops or nested loops based on the loop dependency graph model, retiming, and ...

**Keywords:** code size, embedded DSP, loop fusion, retiming, scheduling

14 Loop optimization for a class of memory-constrained computations



D. Cociorva, J. W. Wilkins, C. Lam, G. Baumgartner, J. Ramanujam, P. Sadayappan

June 2001 **Proceedings of the 15th international conference on Supercomputing ICS '01**

Publisher: ACM Press

Full text available: pdf(160.59 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Compute-intensive multi-dimensional summations that involve products of several arrays arise in the modeling of electronic structure of materials. Sometimes several alternative formulations of a computation, representing different space-time trade-offs, are possible. By computing and storing some intermediate arrays, reduction of the number of arithmetic operations is possible, but the size of intermediate temporary arrays may be prohibitively large. Loop fusion can be applied to reduce memor ...

15 A preprocessing step for global loop transformations for data transfer optimization



Koen Danckaert, Francky Catthoor, Hugo De Man

November 2000 **Proceedings of the 2000 international conference on Compilers, architecture, and synthesis for embedded systems CASES '00**

Publisher: ACM Press

Full text available: pdf(202.94 KB) Additional Information: [full citation](#), [citations](#)

16 Integrated Loop Optimizations for Data Locality Enhancement of Tensor Contraction Expressions

Swarup Kumar Sahoo, Sriram Krishnamoorthy, Rajkiran Panuganti, P. Sadayappan

November 2005 **Proceedings of the 2005 ACM/IEEE conference on Supercomputing SC '05**

Publisher: IEEE Computer Society

Full text available: pdf(521.43 KB) Additional Information: [full citation](#), [abstract](#), [index terms](#)


A very challenging issue for optimizing compilers is the phase ordering problem: In what order should a collection of compiler optimizations be performed? We address this problem in the context of optimizing a sequence of tensor contractions. The pertinent loop transformations are loop permutation, tiling, and fusion; in addition, the placement of disk I/O statements crucially affects performance. The space of possible combinations is exponentially large. We develop novel pruning strategies wher ...

17 Loop Alignment for Memory Accesses Optimization

Antoine Fraboulet, Guillaume Huard, Anne Mignotte

November 1999 **Proceedings of the 12th international symposium on System synthesis ISSS '99**

Publisher: IEEE Computer Society

Full text available:  pdf(133.51 KB)Additional Information: [full citation](#), [abstract](#), [citations](#) [Publisher Site](#)


Portable or embedded systems allow more and more complex applications like multimedia today. These applications and submicronic technologies have made the power consumption criterium crucial. We propose new techniques thanks to which we can optimize the behavioral description of an integrated system before the hardware/software partitioning (Cocesign). These transformations are performed on "for" loops that constitute the main parts of the multimedia code which handle the arrays. We present in t ...

18 A transformation-based approach to optimizing loops in database programming languages

Daniel F. Lieuwen, David J. DeWitt

June 1992 **ACM SIGMOD Record , Proceedings of the 1992 ACM SIGMOD international conference on Management of data SIGMOD '92**, Volume 21 Issue 2

Publisher: ACM Press

Full text available:  pdf(1.21 MB)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Database programming languages like O2, E, and O++ include the ability to iterate through a set. Nested iterators can be used to express joins. This paper describes compile-time optimizations similar to relational transformations like join reordering for such programming constructs. This paper also shows how to use a standard transformation-based optimizer to optimize these joins. An optimizer built using the EXODUS Opt ...

19 Handling irreducible loops: optimized node splitting versus DJ-graphs

Sebastian Unger, Frank Mueller

July 2002 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 24 Issue 4

Publisher: ACM Press

Full text available:  pdf(386.11 KB)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper addresses the question of how to handle irreducible regions during optimization, which has become even more relevant for contemporary processors since recent VLIW-like architectures highly rely on instruction scheduling. The contributions of this paper are twofold. First, a method of optimized node splitting to transform irreducible regions of control flow into reducible regions is formally defined and its correctness is shown. This method is superior to approaches previously publishe ...


**Keywords:** Code optimization, compilation, control flow graphs, instruction-level parallelism, irreducible flowgraphs, loops, node splitting, reducible flowgraphs

20 Flattening and parallelizing irregular, recurrent loop nests

Anwar M. Ghuloum, Allan L. Fisher

August 1995 **ACM SIGPLAN Notices , Proceedings of the fifth ACM SIGPLAN symposium on Principles and practice of parallel programming PPOPP '95**, Volume 30 Issue 8

Publisher: ACM Press

Full text available:  pdf(1.17 MB)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Irregular loop nests in which the loop bounds are determined dynamically by indexed arrays are difficult to compile into expressive parallel constructs, such as segmented scans and reductions. In this paper, we describe a suite of transformations to automatically parallelize such irregular loop nests, even in the presence of recurrences. We describe a simple, general loop flattening transformation, along with new optimizations which make it a viable compiler transformation. A robust recurrence ...

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.  
[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)


[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

 Search: ☒ The ACM Digital Library ☐ The Guide

+nest\* +loop +compil\*



THE ACM DIGITAL LIBRARY


[Feedback](#) [Report a problem](#) [Satisfaction survey](#)
Terms used: **nest loop compil**

Found 5,345 of 215,737

Sort results by

relevance

Display results

expanded form

[Save results to a Binder](#)[Search Tips](#)[Open results in a new window](#)[Try an Advanced Search](#)[Try this search in The ACM Guide](#)

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance scale ☐ ☐ ☐ ☐ ☐**1** [Compile-time minimisation of load imbalance in loop nests](#)

Rizos Sakellariou, John R. Gurd

July 1997 **Proceedings of the 11th international conference on Supercomputing ICS '97**

Publisher: ACM Press

Full text available: [pdf\(997.11 KB\)](#) Additional Information: [full citation](#), [references](#), [index terms](#)**2** [A compiler algorithm for optimizing locality in loop nests](#)

M. Kandemir, J. Ramanujam, A. Choudhary

July 1997 **Proceedings of the 11th international conference on Supercomputing ICS '97**

Publisher: ACM Press

Full text available: [pdf\(1.08 MB\)](#) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)**3** [Compiler techniques for data synchronization in nested parallel loops](#)

Peiyi Tang, Pen-Chung Yew, Chuan-Qi Zhu

June 1990 **ACM SIGARCH Computer Architecture News , Proceedings of the 4th international conference on Supercomputing ICS '90**, Volume 18 Issue 3b

Publisher: ACM Press

Full text available: [pdf\(1.19 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The major source of parallelism in ordinary programs is do loops. When loop iterations of parallelized loops are executed on multiprocessors, the cross-iteration data dependencies need to be enforced by synchronization between processors. Existing data synchronization schemes are either too simple to handle general nested loop structures with non-trivial array subscript functions or inefficient due to the large run-time overhead. In this paper, we propose a new synchronization sch ...

**4** [Optimal tile size adjustment in compiling general DOACROSS loop nests](#)

Hiroshi Ohta, Yasuhiko Saito, Masahiro Kainaga, Hiroyuki Ono

July 1995 **Proceedings of the 9th international conference on Supercomputing ICS '95**

Publisher: ACM Press

Full text available: [pdf\(822.07 KB\)](#) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

5 Flattening and parallelizing irregular, recurrent loop nests

Anwar M. Ghuloum, Allan L. Fisher

August 1995 **ACM SIGPLAN Notices , Proceedings of the fifth ACM SIGPLAN symposium on Principles and practice of parallel programming PPOPP '95**, Volume 30 Issue 8

Publisher: ACM Press

Full text available: pdf(1.17 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Irregular loop nests in which the loop bounds are determined dynamically by indexed arrays are difficult to compile into expressive parallel constructs, such as segmented scans and reductions. In this paper, we describe a suite of transformations to automatically parallelize such irregular loop nests, even in the presence of recurrences. We describe a simple, general loop flattening transformation, along with new optimizations which make it a viable compiler transformation. A robust recurrence ...

6 Tiling imperfectly-nested loop nests

Nawaaz Ahmed, Nikolay Mateev, Keshav Pingali

November 2000 **Proceedings of the 2000 ACM/IEEE conference on Supercomputing (CDROM) Supercomputing '00**

Publisher: IEEE Computer Society

Full text available: pdf(205.52 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)[Publisher Site](#)

Tiling is one of the more important transformations for enhancing locality of reference in programs. Intuitively, tiling a set of loops achieves the effect of interleaving iterations of these loops. Tiling of perfectly-nested loop nests (which are loop nests in which all assignment statements are contained in the innermost loop) is well understood. In practice, many loop nests are imperfectly nested, so existing compilers use heuristics to try to find a sequence of transformations that can ...

7 A compiler optimization to reduce execution time of loop nest

Oh-Young Kwon, Gi-Ho Park, Tack-Don Han

March 1996 **ACM SIGARCH Computer Architecture News**, Volume 24 Issue 1

Publisher: ACM Press

Full text available: pdf(333.41 KB)

Additional Information: [full citation](#), [abstract](#), [index terms](#)

In this paper, a compiler optimization to reduce the execution time of loop nest is proposed. Loop tiling is used to optimize loop nest. Loop tiling is the well-known optimization for improving locality. However, it has a count result that increases the number of instructions to loop control. These increased instructions disturb the effect of locality optimization. Therefore, the tiling of innermost loop is not performed in order to reduce the instructions for loop control in this paper. This optimization ...

8 Optimized unrolling of nested loops

Vivek Sarkar

May 2000 **Proceedings of the 14th international conference on Supercomputing ICS '00**

Publisher: ACM Press

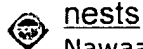
Full text available: pdf(1.10 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

In this paper, we address the problems of automatically selecting unroll factors for perfectly nested loops, and generating compact code for the selected unroll factors. Compared to past work, the contributions of our work include a) a more detailed cost model that includes ILP and 1-cache considerations, b) a new code generation algorithm for unrolling nested loops that generates more compact code (with fewer remainder loops) than the unroll-and-jam transformation ...

## 9

Synthesizing transformations for locality enhancement of imperfectly-nested loop

**nests**

Nawaaz Ahmed, Nikolay Mateev, Keshav Pingali

May 2000 **Proceedings of the 14th international conference on Supercomputing ICS '00****Publisher:** ACM PressFull text available: [pdf\(1.06 MB\)](#)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We present an approach for synthesizing transformations to enhance locality in imperfectly-nested loops. The key idea is to embed the iteration space of every statement in a loop nest into a special iteration space called the product space. The product space can be viewed as a perfectly-nested loop nest, so embedding generalizes techniques like code sinking and loop fusion that are used in ad hoc ways in current compilers to produce perfectly-nested loops from imperfectly-n ...

**10 Parallel and distributed systems (PDS): Automatic parallel code generation for tiled****nested loops**

Georgios Goumas, Nikolaos Drosinos, Maria Athanasaki, Nectarios Koziris

March 2004 **Proceedings of the 2004 ACM symposium on Applied computing SAC '04****Publisher:** ACM PressFull text available: [pdf\(367.02 KB\)](#)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

This paper presents an overview of our work, concerning a complete end-to-end framework for automatically generating message passing parallel code for tiled nested for-loops. It considers general parallelepiped tiling transformations and general convex iteration spaces. We address all problems regarding both the generation of sequential tiled code and its parallelization. We have implemented our techniques in a tool which automatically generates MPI parallel code and conducted several series of ...

**Keywords:** MPI, automatic SPMD code generation, nested loops, parallelizing compilers, supernodes, tiling

**11 Compiler analysis and optimization: General loop fusion technique for nested loops****considering timing and code size**

Meilin Liu, Qingfeng Zhuge, Zili Shao, Edwin H.-M. Sha

September 2004 **Proceedings of the 2004 international conference on Compilers, architecture, and synthesis for embedded systems CASES '04****Publisher:** ACM PressFull text available: [pdf\(307.40 KB\)](#)Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Loop fusion is commonly used to improve the instruction-level parallelism of loops for high-performance embedded computing systems. Loop fusion, however, is not always directly applicable because the fusion prevention dependencies may exist among loops. Most of the existing techniques still have limitations in fully exploiting the advantages of loop fusion. In this paper, we present a general loop fusion technique for loops or nested loops based on the loop dependency graph model, retiming, and ...

**Keywords:** code size, embedded DSP, loop fusion, retiming, scheduling

**12 Compiling nested data-parallel programs for shared-memory multiprocessors**

Siddhartha Chatterjee

July 1993 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 15 Issue 3**Publisher:** ACM PressFull text available: [pdf\(4.17 MB\)](#)Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#), [review](#)

**Keywords:** compilers, data parallelism, shared-memory multiprocessors

13 Compilers and Optimization: Combined partitioning and data padding for scheduling multiple loop nests



Zhong Wang, Edwin H.-M. Sha, Xiaobo (Sharon) Hu

November 2001 **Proceedings of the 2001 international conference on Compilers, architecture, and synthesis for embedded systems CASES '01**

**Publisher:** ACM Press

Full text available: pdf(166.83 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

With the widening performance gap between processors and main memory, efficient memory accessing behavior is necessary for good program performance. Loop partition is an effective way to exploit the data locality. Traditional loop partition techniques, however, consider only a singleton nested loop. This paper presents multiple loop partition scheduling technique, which combines the loop partition and data padding to generate the detailed partition schedule. The computation and data prefetching ...

14 Removal of invariant statements from nested-loops in a single effective compiler pass



D. Neel, M. Amirchahy

January 1975 **ACM SIGPLAN Notices , Proceedings of the conference on Programming languages and compilers for parallel and vector machines**, Volume 10 Issue 3

**Publisher:** ACM Press

Full text available: pdf(636.87 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This document presents how one of the most important optimizations that a program may undergo is dealt with by means of attributes [7]. A semantic formalization of the classical method which consists of removing all loop-independent statements from the articulation blocks of a loop is given. The method is equally well applicable to algebraic languages or their intermediate code : in a high level language even very well constructed programs quite often contain in their intermediate code vers ...

**Keywords:** Articulation block, Basic block, Compiler optimization, Evaluation program, Invariant statement, Semantic attributes

15 Quantifying loop nest locality using SPEC'95 and the perfect benchmarks



Kathryn S. McKinley, Olivier Temam

November 1999 **ACM Transactions on Computer Systems (TOCS)**, Volume 17 Issue 4

**Publisher:** ACM Press

Full text available: pdf(635.63 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This article analyzes and quantifies the locality characteristics of numerical loop nests in order to suggest future directions for architecture and software cache optimizations. Since most programs spend the majority of their time in nests, the vast majority of cache optimization techniques target loop nests. In contrast, the locality characteristics that drive these optimizations are usually collected across the entire application rather than at the nest level. Researchers have studied nu ...

16 Compilers, supercomputing and quantum computing: A general approach for partitioning N-dimensional parallel nested loops with conditionals




Arun Kejariwal, Alexandru Nicolau, Hideki Saito, Xinmin Tian, Milind Girkar, Utpal Banerjee, Constantine D. Polychronopoulos

July 2006 **Proceedings of the eighteenth annual ACM symposium on Parallelism in algorithms and architectures SPAA '06**

**Publisher:** ACM Press

Full text available: Additional Information:



 [pdf\(532.66 KB\)](#)[full citation](#), [abstract](#), [references](#), [index terms](#)

Parallel loops account for the greatest amount of parallelism in scientific and numerical codes. For example, most of the **DO** loops in SPEC CFP2000 and SPEC OMPM2001 are of **DOALL** type and account for a large percentage of the total execution time. One of the ways to exploit parallelism is to partition the iteration space of a **DOALL** loop amongst different processors in a parallel processor system. Naturally, a good partitioning is of key importance to achieve high performance ...

**Keywords:** Fourier-Motzkin elimination, affine, conditionals, parallel loops, partitioning


### 17 [A hyperplane based approach for optimizing spatial locality in loop nests](#)



M. Kandemir, A. Choudhary, N. Shenoy, P. Banerjee, J. Ramanujam

July 1998 **Proceedings of the 12th international conference on Supercomputing ICS '98**

**Publisher:** ACM Press

Full text available:  [pdf\(1.13 MB\)](#)

Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

### 18 [Automatic memory layout transformations to optimize spatial locality in parameterized loop nests](#)



Philippe Clauss, Benoît Meister

March 2000 **ACM SIGARCH Computer Architecture News**, Volume 28 Issue 1

**Publisher:** ACM Press

Full text available:  [pdf\(537.80 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [index terms](#)

One of the most efficient ways to improve program performances onto nowadays computers is to optimize the way cache memories are used. In particular, many scientific applications contain loop nests that operate on large multi-dimensional arrays whose sizes are often parameterized. No special attention is paid to cache memory performance when such loops are written. In this work, we focus on spatial locality optimization such that all the data that are loaded as a block in the cache will be used ...

**Keywords:** Ehrhart polynomials, cache memory, loop nests, optimizing compiler, parameterized polyhedron, program performance optimization, spatial locality

### 19 [Control Flow Driven Splitting of Loop Nests at the Source Code Level](#)



Heiko Falk, Peter Marwedel

March 2003 **Proceedings of the conference on Design, Automation and Test in Europe - Volume 1 DATE '03**

**Publisher:** IEEE Computer Society

Full text available:  [pdf\(215.57 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [index terms](#)



[Publisher Site](#)


This paper presents a novel source code transformation for control flow optimization called loop nest splitting which minimizes the number of executed if-statements in loop nests of embedded multimedia applications. The goal of the optimization is to reduce runtimes and energy consumption. The analysis techniques are based on precise mathematical models combined with genetic algorithms. Due to the inherent portability of source code transformations, a very detailed benchmarking using 10 differen ...

### 20 [A quantitative analysis of loop nest locality](#)



Kathryn S. McKinley, Olivier Temam

September 1996 **ACM SIGPLAN Notices**, **ACM SIGOPS Operating Systems Review**, **Proceedings of the seventh international conference on Architectural support for programming languages and operating systems ASPLOS-VII**, Volume 31, 30 Issue 9, 5

**Publisher:** ACM PressFull text available:  pdf(1.49 MB)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper analyzes and quantifies the locality characteristics of numerical loop nests in order to suggest future directions for architecture and software cache optimizations. Since most programs spend the majority of their time in nests, the vast majority of cache optimization techniques target loop nests. In contrast, the locality characteristics that drive these optimizations are usually collected across the entire application rather than the nest level. Indeed, researchers have studied nume ...

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)


[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

 Search: ☒ The ACM Digital Library ☐ The Guide

+critical +code +compil\*


[Feedback](#) [Report a problem](#) [Satisfaction survey](#)
Terms used: **critical code compil**

Found 11,404 of 215,737

Sort results by

relevance

Display results

expanded form

☒ [Save results to a Binder](#)
☒ [Search Tips](#)
☐ Open results in a new window

[Try an Advanced Search](#)
[Try this search in The ACM Guide](#)

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance scale ☐ ☐ ☐ ☐ ☐

### 1 [A study of source-level compiler algorithms for automatic construction of pre-execution code](#)

Dongkeun Kim, Donald Yeung

August 2004 **ACM Transactions on Computer Systems (TOCS)**, Volume 22 Issue 3

Publisher: ACM Press

Full text available: [pdf\(1.55 MB\)](#)
 Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Pre-execution is a promising latency tolerance technique that uses one or more helper threads running in spare hardware contexts ahead of the main computation to trigger long-latency memory operations early, hence absorbing their latency on behalf of the main computation. This article investigates several source-to-source C compilers for extracting pre-execution thread code automatically, thus relieving the programmer or hardware from this onerous task. We present an aggressive profile-driven co ...

**Keywords:** Data prefetching, memory-level parallelism, multithreading, pre-execution, prefetch conversion, program slicing, speculative loop parallelization

### 2 [Code optimization - I: Compiler optimization-space exploration](#)

Spyridon Triantafyllis, Manish Vachharajani, Neil Vachharajani, David I. August

March 2003 **Proceedings of the international symposium on Code generation and optimization: feedback-directed and runtime optimization CGO '03**

Publisher: IEEE Computer Society

Full text available: [pdf\(1.19 MB\)](#)
 Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

To meet the demands of modern architectures, optimizing compilers must incorporate an ever larger number of increasingly complex transformation algorithms. Since code transformations may often degrade performance or interfere with subsequent transformations, compilers employ predictive heuristics to guide optimizations by predicting their effects a priori. Unfortunately, the unpredictability of optimization interaction and the irregularity of today's wide-issue machines severely limit the accuracy ...

### 3 [Compiler optimization of scalar value communication between speculative threads](#)

Antonia Zhai, Christopher B. Colohan, J. Gregory Steffan, Todd C. Mowry

October 2002 **ACM SIGPLAN Notices**, **ACM SIGOPS Operating Systems Review**, **ACM SIGARCH Computer Architecture News**, **Proceedings of the 10th international conference on Architectural support for programming languages and operating systems ASPLOS-X**, Volume 37, 36, 30 Issue 10, 5, 5

Publisher: ACM

Full text available:  [pdf\(1.39 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [cited by](#)

While there have been many recent proposals for hardware that supports *Thread-Level Speculation* (TLS), there has been relatively little work on compiler optimizations to fully exploit this potential for parallelizing programs optimistically. In this paper, we focus on one important limitation of program performance under TLS, which is stalls due to forwarding scalar values between threads that would otherwise cause frequent data dependences. We present and evaluate dataflow algorithms for ...


#### 4 Compiling real-time programs into schedulable code



Seongsoo Hong, Richard Gerber

June 1993 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1993 conference on Programming language design and implementation PLDI '93**, Volume 28  
Issue 6

**Publisher:** ACM Press

Full text available:  [pdf\(1.06 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We present a programming language with first-class timing constructs, whose semantics is based on time-constrained relationships between observable events. Since a system specification postulates timing relationships between events, realizing the specification in a program becomes a more straightforward process. Using these constraints, as well as those imposed by data and control flow properties, our objective is to transform the code so that its worst-case execution time is con ...


#### 5 Code management: HotpathVM: an effective JIT compiler for resource-constrained devices



Andreas Gal, Christian W. Probst, Michael Franz

June 2006 **Proceedings of the 2nd international conference on Virtual execution environments VEE '06**

**Publisher:** ACM Press

Full text available:  [pdf\(221.33 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

We present a just-in-time compiler for a Java VM that is small enough to fit on resource-constrained devices, yet is surprisingly effective. Our system dynamically identifies traces of frequently executed bytecode instructions (which may span several basic blocks across several methods) and compiles them via Static Single Assignment (SSA) construction. Our novel use of SSA form in this context allows to hoist instructions across trace side-exits without necessitating expensive compensation code ...

**Keywords:** dynamic compilation, embedded and resource-constrained systems, mixed-mode interpretive compiled systems, software trace scheduling, static single assignment form, virtual machines

#### 6 Short presentations with posters: SCCP/x: a compilation profile to support testing and verification of optimized code



Raimund Kirner

September 2007 **Proceedings of the 2007 international conference on Compilers, architecture, and synthesis for embedded systems CASES '07**

**Publisher:** ACM Press

Full text available:  [pdf\(350.81 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Embedded systems are often used in safety-critical environments. Thus, thorough testing of them is mandatory. A quite active research area is the automatic test-case generation for testing embedded systems. To achieve high retargetability of the testing framework, the test-case generation has to be done at source-code level. However, it is challenging to guarantee that the test-cases obtained from the source code are also valid at the object-code level, since even in safety-critical domains pr ...

**Keywords:** code transformation, compiler, decision coverage, optimization, structural code-coverage preservation, testing

7 Exploiting parallel microprocessor microarchitectures with a compiler code generator

W. W. Hwu, P. P. Chang

May 1988 **ACM SIGARCH Computer Architecture News , Proceedings of the 15th Annual International Symposium on Computer architecture ISCA '88**,  
Volume 16 Issue 2**Publisher:** IEEE Computer Society Press, ACM PressFull text available: pdf(890.51 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

With advances in VLSI technology, microprocessor designers can provide more microarchitectural parallelism to increase performance. We have identified four major forms of such parallelism: multiple microoperations issued per cycle, multiple result distribution buses, multiple execution units, and pipelined execution units. The experiments reported in this paper address two important issues: The effects of these forms and the appropriate balance among them. A central microar ...

8 Design and evaluation of compiler algorithms for pre-execution

Dongkeun Kim, Donald Yeung

October 2002 **ACM SIGPLAN Notices , ACM SIGOPS Operating Systems Review , ACM SIGARCH Computer Architecture News , Proceedings of the 10th international conference on Architectural support for programming languages and operating systems ASPLOS-X**, Volume 37 , 36 , 30 Issue 10 , 5 , 5**Publisher:** ACMFull text available: pdf(1.43 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [cited by](#)

Pre-execution is a promising latency tolerance technique that uses one or more helper threads running in spare hardware contexts ahead of the main computation to trigger long-latency memory operations early, hence absorbing their latency on behalf of the main computation. This paper investigates a source-to-source C compiler for extracting pre-execution thread code automatically, thus relieving the programmer or hardware from this onerous task. At the heart of our compiler are three algorithms. ...

9 C and tcc: a language and compiler for dynamic code generation

Massimiliano Poletto, Wilson C. Hsieh, Dawson R. Engler, M. Frans Kaashoek

March 1999 **ACM Transactions on Programming Languages and Systems (TOPLAS)**,  
Volume 21 Issue 2**Publisher:** ACM PressFull text available: pdf(471.68 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Dynamic code generation allows programmers to use run-time information in order to achieve performance and expressiveness superior to those of static code. The 'C(Tick C) language is a superset of ANSI C that supports efficient and high-level use of dynamic code generation. 'C provides dynamic code generation at the level of C expressions and statements and supports the composition of dynamic code at run time. These features enable programmers to add dynamic code generation ...

**Keywords:** ANSI C, compilers, dynamic code generation, dynamic code optimization10 Code compression: Compiler optimization and ordering effects on VLIW code compression

Montserrat Ros, Peter Sutton

October 2003 **Proceedings of the 2003 international conference on Compilers, architecture and synthesis for embedded systems CASES '03****Publisher:** ACM PressFull text available: pdf(334.18 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Code size has always been an important issue for all embedded applications as well as

larger systems. Code compression techniques have been devised as a way of battling bloated code; however, the impact of VLIW compiler methods and outputs on these compression schemes has not been thoroughly investigated. This paper describes the application of single- and multiple-instruction dictionary methods for code compression to decrease overall code size for the TI TMS320C6xxx DSP family. The compression ...

**Keywords:** VLIW, code compression, compiler optimizations

# 11 Compilation: Efficient spill code for SDRAM



V. Krishna Nandivada, Jens Palsberg

October 2003 **Proceedings of the 2003 international conference on Compilers, architecture and synthesis for embedded systems CASES '03**

**Publisher:** ACM Press

Full text available: [pdf\(199.32 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Processors such as StrongARM and memory such as SDRAM enable efficient execution of multiple loads and stores in a single instruction. This is particularly useful in connection with register allocation where spill code may need to save and restore multiple registers. Until now, there has been no effective strategy for utilizing this to its full potential. In this paper we investigate the use of SDRAM for optimization of spill code. The core of the problem is to arrange the variables in the spill ...

**Keywords:** SDRAM, integer linear programming, memory layout, optimization

# 12 Speculative disambiguation: a compilation technique for dynamic memory disambiguation

A. S. Huang, G. Slavenburg, J. P. Shen

April 1994 **ACM SIGARCH Computer Architecture News , Proceedings of the 21st annual international symposium on Computer architecture ISCA '94**, Volume 22 Issue 2

**Publisher:** IEEE Computer Society Press, ACM

Full text available: [pdf\(1.09 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [cited by](#), [index terms](#)

Ambiguous memory references have always been one of the main sources of performance bottlenecks. Many papers have addressed this problem using static disambiguation. These methods work extremely well when the memory access pattern is linear and predictable. However they are ineffective when the memory access pattern is nonlinear or when the access pattern cannot be determined statically. For these difficult problems, this paper presents speculative disambiguation, a compilation technique for arc ...

# 13 Avoidance and suppression of compensation code in a trace scheduling compiler



Stefan M. Freudenberger, Thomas R. Gross, P. Geoffrey Lowney

July 1994 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 16 Issue 4

**Publisher:** ACM Press

Full text available: [pdf\(3.58 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Trace scheduling is an optimization technique that selects a sequence of basic blocks as a trace and schedules the operations from the trace together. If an operation is moved across basic block boundaries, one or more compensation copies may be required in the off-trace code. This article discusses the generation of compensation code in a trace scheduling compiler and presents techniques for limiting the amount of compensation code: avoidance (restricting code motion so that no compensatio ...

**Keywords:** SPEC89, instruction-level parallelism, performance evaluation, trace scheduling

14 Shangri-La: achieving high performance from compiled network applications while enabling ease of programming



Michael K. Chen, Xiao Feng Li, Ruiqi Lian, Jason H. Lin, Lixia Liu, Tao Liu, Roy Ju

June 2005 **ACM SIGPLAN Notices , Proceedings of the 2005 ACM SIGPLAN conference on Programming language design and implementation PLDI '05**, Volume 40  
Issue 6

**Publisher:** ACM Press

Full text available: [pdf\(480.93 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Programming network processors is challenging. To sustain high line rates, network processors have extremely tight memory access and instruction budgets. Achieving desired performance has traditionally required hand-coded assembly. Researchers have recently proposed high-level programming languages for packet processing, but the challenges of compiling these languages into code that is competitive with hand-tuned assembly remain unanswered. This paper describes the Shangri-La compiler, which achieves ...

**Keywords:** chip multiprocessors, dataflow programming, network processors, packet processing, program partitioning, throughput-oriented computing

15 Embedded systems: applications, solutions and techniques (EMBS): Selective compilation via fast code analysis and bytecode tracing



G. Agosta, S. Crespi Reghizzi, P. Palumbo, M. Sykora

April 2006 **Proceedings of the 2006 ACM symposium on Applied computing SAC '06**

**Publisher:** ACM Press

Full text available: [pdf\(265.02 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Modern Java Virtual Machines (JVM) commonly adopt Just-In-Time (JIT) compilation to speed up the execution of Java Bytecode. However, the effort of compiling a region of code is only worth if the code is frequently executed. Therefore, *Selective Compilation* is employed so that the JIT compiler is only invoked on those regions of code where most of the computation is performed (hot spots). The core task in Selective Compilation is to correctly identify the hot spots in a program. In our Se ...

16 Compilation/code generation: A simplified java bytecode compilation system for resource-constrained embedded processors



Carmen Badea, Alexandru Nicolau, Alexander V. Veidenbaum

September 2007 **Proceedings of the 2007 international conference on Compilers, architecture, and synthesis for embedded systems CASES '07**

**Publisher:** ACM Press

Full text available: [pdf\(439.70 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Embedded platforms are resource-constrained systems in which performance and memory requirements of executed code are of critical importance. However, standard techniques such as full just-in-time (JIT) compilation and/or adaptive optimization (AO) may not be appropriate for this type of systems due to memory and compilation overheads.

The research presented in this paper proposes a technique that combines some of the main benefits of JIT compilation, superoperators (SOs) and profile-guide ...

**Keywords:** adaptive optimization, embedded systems, java virtual machine, profile-guided optimization, superoperators

17 Formal certification of a compiler back-end or: programming a compiler with a proof assistant



Xavier Leroy

January 2006 **ACM SIGPLAN Notices , Conference record of the 33rd ACM SIGPLAN-SIGACT symposium on Principles of programming languages POPL '06**,  
Volume 41 Issue 1

**Publisher:** ACM Press

Full text available:  [pdf\(187.24 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper reports on the development and formal certification (proof of semantic preservation) of a compiler from Cminor (a C-like imperative language) to PowerPC assembly code, using the Coq proof assistant both for programming the compiler and for proving its correctness. Such a certified compiler is useful in the context of formal methods applied to the certification of critical software: the certification of the compiler guarantees that the safety properties proved on the source code hold f ...

**Keywords:** certified compilation, compiler transformations and optimizations, program proof, semantic preservation, the Coq theorem prover

18 Virtual machines and compilation: Implementing fast JVM interpreters using Java itself



Michael Bebenita, Andreas Gal, Michael Franz

September 2007 **Proceedings of the 5th international symposium on Principles and practice of programming in Java PPPJ '07**

**Publisher:** ACM Press

Full text available:  [pdf\(741.09 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Most Java Virtual Machines (JVMs) are themselves written in unsafe languages, making it unduly difficult to build trustworthy and safe JVM platforms. While some progress has been made on removing *compilers* from the trusted computing base (using certifying compilation), JVM *interpreters* continue to be built almost exclusively in C/C++. We have implemented an alternative approach, in which the JVM interpreter itself is built in Java, and runs atop a host JVM execution environment. ...


**Keywords:** Java virtual machine, interpreter design, metacircular/self-interpreters, minimal trusted computing base

19 Critical path reduction for scalar programs

Michael Schlansker, Vinod Kathail

December 1995 **Proceedings of the 28th annual international symposium on Microarchitecture MICRO 28**

**Publisher:** IEEE Computer Society Press

Full text available:  [pdf\(1.38 MB\)](#)

Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

20 Building Intrusion-Tolerant Secure Software

Tao Zhang, Xiaotong Zhuang, Santosh Pande

March 2005 **Proceedings of the international symposium on Code generation and optimization CGO '05**

**Publisher:** IEEE Computer Society

Full text available:  [pdf\(234.98 KB\)](#)





Additional Information: [full citation](#), [abstract](#), [index terms](#)

In this work, we develop a secret sharing based compiler solution to achieve confidentiality, integrity and availability (intrusion tolerance) of critical data together, rather than tackling them one by one as in previous approaches. Under our scheme, some critical data values are automatically identified by the compiler, whereas some others are specified by the user. The compiler generates code for scattering/assembling and verifying of those critical data values using secret sharing scheme. In ...



The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)


[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

 Search: ☒ The ACM Digital Library ☐ The Guide


[Feedback](#) [Report a problem](#) [Satisfaction survey](#)
Terms used: **critical code function**Found **20,242** of **215,737**

Sort results by

Display results

[Save results to a Binder](#)[Search Tips](#)
☐ Open results in a new window
[Try an Advanced Search](#)[Try this search in The ACM Guide](#)

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance scale ☐ ☐ ☐ ☐ ☐

### 1 [Specification and architecture challenges in high-level synthesis: A code refinement methodology for performance-improved synthesis from C](#)



Greg Stitt, Frank Vahid, Walid Najjar

 November 2006 **Proceedings of the 2006 IEEE/ACM international conference on Computer-aided design ICCAD '06**

Publisher: ACM Press

Full text available: [pdf\(219.07 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Although many recent advances have been made in hardware synthesis techniques from software programming languages such as C, the performance of synthesized hardware commonly suffers due to the use of C constructs and coding practices that are not appropriate for hardware. Most previous approaches to addressing this problem require drastic changes to coding practice. We present an approach that instead requires only minimal changes but yields significant speedups. In this approach, a software ...

**Keywords:** FPGA, code refinement, coding guidelines, compilation, embedded systems, hardware/software partitioning, synthesis

### 2 [Inline function expansion for compiling C programs](#)



P. P. Chang, W.-W. Hwu

 June 1989 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1989 Conference on Programming language design and implementation PLDI '89**, Volume 24 Issue 7

Publisher: ACM Press

Full text available: [pdf\(1.14 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Inline function expansion replaces a function call with the function body. With automatic inline function expansion, programs can be constructed with many small functions to handle complexity and then rely on the compilation to eliminate most of the function calls. Therefore, inline expansion serves a tool for satisfying two conflicting goals: minimizing the complexity of the program development and minimizing the function call overhead of program execution. A simple inline expansion procedure ...

### 3 [Function unit specialization through code analysis](#)



Daniel Benyamin, William H. Mangione-Smith

 November 1999 **Proceedings of the 1999 IEEE/ACM international conference on Computer-aided design ICCAD '99**

Publisher: IEEE Press

Full text available: [pdf\(105.29 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Many previous attempts at ASIP synthesis have employed template matching techniques


to target function units to application code, or directly design new units to extract maximum performance. This paper presents an entirely new approach to specializing hardware for application specific needs. In our framework of a parameterized VLIW processor, we use a post-modulo scheduling analysis to reduce the allocated hardware resources while increasing the code's performance. Initial results i ...

#### 4 INSIDE: INstruction Selection/Identification & Design Exploration for Extensible Processors

Newton Cheung, Sri Parameswaran, Jörg Henkel

November 2003 **Proceedings of the 2003 IEEE/ACM international conference on Computer-aided design ICCAD '03**

**Publisher:** IEEE Computer Society

Full text available:  pdf(248.23 KB) Additional Information: [full citation](#), [abstract](#), [citations](#), [index terms](#)

This paper presents the INSIDE system that rapidly searches the design space for extensible processors, given area and performance constraints of an embedded application, while minimizing the design turn-around-time. Our system consists of a) a methodology to determine which code segments are most suited for implementation as a set of extensible instructions, b) a heuristic algorithm to select pre-configured extensible processors as well as extensible instructions (library), and c) an estimation tool ...

#### 5 EPIC compilation: Inlining of mathematical functions in HP-UX for Itanium® 2

James W. Thomas

March 2003 **Proceedings of the international symposium on Code generation and optimization: feedback-directed and runtime optimization CGO '03**

**Publisher:** IEEE Computer Society

Full text available:  pdf(783.82 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

HP-UX compilers inline mathematical functions for Itanium Processor Family (IPF) systems to improve throughput 4X-8X versus external library calls, achieving speeds comparable to highly tuned vector functions, without requiring the user to code for a vector interface and without sacrificing accuracy or edge-case behaviors. This paper highlights IPF architectural features that support implementation of high-performance, high-quality math functions for inlining. It discusses strategies for utilizing ...


#### 6 Survey of code-size reduction methods



Árpád Beszédes, Rudolf Ferenc, Tibor Gyimóthy, André Dolenc, Konsta Karsisto

September 2003 **ACM Computing Surveys (CSUR)**, Volume 35 Issue 3

**Publisher:** ACM Press

Full text available:  pdf(443.89 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Program code compression is an emerging research activity that is having an impact in several production areas such as networking and embedded systems. This is because the reduced-sized code can have a positive impact on network traffic and embedded system costs such as memory requirements and power consumption. Although code-size reduction is a relatively new research area, numerous publications already exist on it. The methods published usually have different motivations and a variety of applications ...

**Keywords:** code compaction, code compression, method assessment, method evaluation

#### 7 Low Power Embedded Software Optimization Using Symbolic Algebra

A. Peymandoust, T. Simunic, G. de Micheli

March 2002 **Proceedings of the conference on Design, automation and test in Europe DATE '02**

**Publisher:** IEEE Computer Society

Full text available:  pdf(119.80 KB) Additional Information: [full citation](#), [abstract](#), [citations](#)

The market demand for portable multimedia applications has exploded in the recent years. Unfortunately, for such applications current compilers and software optimization methods often require designers to do part of the optimization manually. Specifically, the high-level arithmetic optimizations and the use of complex instructions are left to the designers' ingenuity. In this paper, we present a tool flow, SymSoft, that automates the optimization of power-intensive algorithmic constructs using symbolic a ...

## 8 Compiling real-time programs into schedulable code



Seongsoo Hong, Richard Gerber

June 1993 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1993 conference on Programming language design and implementation PLDI '93**, Volume 28  
Issue 6

**Publisher:** ACM Press

Full text available: [pdf\(1.06 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We present a programming language with first-class timing constructs, whose semantics is based on time-constrained relationships between observable events. Since a system specification postulates timing relationships between events, realizing the specification in a program becomes a more straightforward process. Using these constraints, as well as those imposed by data and control flow properties, our objective is to transform the code so that its worst-case execution time is con ...

## 9 Developing safety critical software for an unmanned aerial vehicle situational awareness tool



Ricky E. Sward, Mark Gerken

November 2006 **Proceedings of the 2006 annual ACM SIGAda international conference on Ada SIGAda '06**

**Publisher:** ACM Press

Full text available: [pdf\(1.16 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

In this paper, we describe our application of the SPARK programming language to the development of flight control software for an Unmanned Aerial Vehicle (UAV). The SPARK language was used during a senior-level software engineering course at the US Air Force Academy. This paper uses the year-long project from this course as an example application of SPARK. The process we used to build an interface between C++ and Ada is discussed along with our experiences with using SPARK.

**Keywords:** SPARK, UAV, formal methods, high integrity, safety critical, unmanned aerial vehicle

## 10 Code optimization - I: Compiler optimization-space exploration



Spyridon Triantafyllis, Manish Vachharajani, Neil Vachharajani, David I. August

March 2003 **Proceedings of the international symposium on Code generation and optimization: feedback-directed and runtime optimization CGO '03**

**Publisher:** IEEE Computer Society

Full text available: [pdf\(1.19 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

To meet the demands of modern architectures, optimizing compilers must incorporate an ever larger number of increasingly complex transformation algorithms. Since code transformations may often degrade performance or interfere with subsequent transformations, compilers employ predictive heuristics to guide optimizations by predicting their effects a priori. Unfortunately, the unpredictability of optimization interaction and the irregularity of today's wide-issue machines severely limit the accuracy ...

## 11 Reconfigurable system: A run-time word-level reconfigurable coarse-grain functional unit for a VLIW processor



Natalino G. Busá, Carles Rodoreda Sala

October 2002 **Proceedings of the 15th international symposium on System Synthesis**  
**ISSS '02**

**Publisher:** ACM Press

Full text available:  [pdf\(492.13 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Nowadays, new DSP applications are offering combined and flexible multimedia and telecom services. VLIW processor architectures, which include dedicated but inflexible functional units, are usually tuned to a single specific application. In order to accelerate a wide range of applications, we propose a VLIW processor containing a novel run-time reconfigurable functional unit (RC-FU). Only a few hundred bits and few cycles are necessary to configure a new coarse-grain operation on the RC-FU unit. ...

**Keywords:** VLIW processors, architectural synthesis, reconfigurable logic


12 Static resource models for code-size efficient embedded processors



Qin Zhao, Bart Mesman, Twan Basten

May 2003 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 2 Issue 2

**Publisher:** ACM Press

Full text available:  [pdf\(651.62 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Due to an increasing need for flexibility, embedded systems embody more and more programmable processors as their core components. Due to silicon area and power considerations, the corresponding instruction sets are often highly encoded to minimize code size for given performance requirements. This has hampered the development of robust optimizing compilers because the resulting irregular instruction set architectures are far from convenient compiler targets. Among other considerations, they int ...

**Keywords:** Static resource models, constraint analysis, convex hull, phase coupling, scheduling


13 Functional Equivalence Checking for Verification of Algebraic Transformations on Array-Intensive Source Code



K. C. Shashidhar, Maurice Bruynooghe, Francky Catthoor, Gerda Janssens

March 2005 **Proceedings of the conference on Design, Automation and Test in Europe - Volume 2 DATE '05**

**Publisher:** IEEE Computer Society

Full text available:  [pdf\(177.24 KB\)](#) Additional Information: [full citation](#), [abstract](#), [index terms](#)

Development of energy and performance-efficient embedded software is increasingly relying on application of complex transformations on the critical parts of the source code. Designers applying such nontrivial source code transformations are often faced with the problem of ensuring functional equivalence of the original and transformed programs. Currently they have to rely on incomplete and time-consuming simulation. Formal automatic verification of the transformed program against the original is ...

14 Embedded software automation: from specification to binary: Complex library mapping for embedded software using symbolic algebra



Armita Peymandoust, Giovanni De Micheli, Tajana Simunic

June 2002 **Proceedings of the 39th conference on Design automation DAC '02**

**Publisher:** ACM Press

Full text available:  [pdf\(95.53 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Embedded software designers often use libraries that have been pre-optimized for a given processor to achieve higher code quality. However, using such libraries in legacy code optimization is nontrivial and typically requires manual intervention. This paper presents a methodology that maps algorithmic constructs of the software specification to a library of complex software elements. This library-mapping step is automated by using symbolic algebra techniques. We illustrate the advantages of our ...

**Keywords:** automated library mapping, computation intensive software, embedded software optimization, polynomial representation, symbolic algebra

15 Vigilante: end-to-end containment of internet worms



Manuel Costa, Jon Crowcroft, Miguel Castro, Antony Rowstron, Lidong Zhou, Lintao Zhang, Paul Barham

October 2005 **ACM SIGOPS Operating Systems Review , Proceedings of the twentieth ACM symposium on Operating systems principles SOSP '05**, Volume 39 Issue 5

**Publisher:** ACM Press

Full text available: [pdf\(329.29 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Worm containment must be automatic because worms can spread too fast for humans to respond. Recent work has proposed network-level techniques to automate worm containment; these techniques have limitations because there is no information about the vulnerabilities exploited by worms at the network level. We propose Vigilante, a new end-to-end approach to contain worms automatically that addresses these limitations. Vigilante relies on collaborative worm detection at end hosts, but does not require ...

**Keywords:** control flow analysis; data flow analysis, self-certifying alerts, worm containment

16 Building Intrusion-Tolerant Secure Software



Tao Zhang, Xiaotong Zhuang, Santosh Pande

March 2005 **Proceedings of the international symposium on Code generation and optimization CGO '05**

**Publisher:** IEEE Computer Society

Full text available: [pdf\(234.98 KB\)](#) Additional Information: [full citation](#), [abstract](#), [index terms](#)

In this work, we develop a secret sharing based compiler solution to achieve confidentiality, integrity and availability (intrusion tolerance) of critical data together, rather than tackling them one by one as in previous approaches. Under our scheme, some critical data values are automatically identified by the compiler, whereas some others are specified by the user. The compiler generates code for scattering/assembling and verifying of those critical data values using secret sharing scheme. In ...

17 Experience with access functions in an experimental compiler



Frederic N. Ris

September 1984 **ACM SIGMICRO Newsletter**, Volume 15 Issue 3

**Publisher:** ACM Press

Full text available: [pdf\(1.12 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#)

This paper describes an access function subsystem embedded in portions of an experimental microcode compiler which was built and used during 1973--6 using the IBM PL/I optimizing compiler under VM/370 and CMS. The use of the access function subsystem in this context was itself an experiment, performed by a group for all of whom PL/I was a new language and VM/370 a new operating system. The implementation of the subsystem was done strictly within the confines of the PL/I language. The basic objec ...

18 Binary synthesis



Greg Stitt, Frank Vahid

August 2007 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volume 12 Issue 3

**Publisher:** ACM Press

Full text available: [pdf\(341.48 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Recent high-level synthesis approaches and C-based hardware description languages attempt to improve the hardware design process by allowing developers to capture

desired hardware functionality in a well-known high-level source language. However, these approaches have yet to achieve wide commercial success due in part to the difficulty of incorporating such approaches into software tool flows. The requirement of using a specific language, compiler, or development environment may cause many so ...

**Keywords:** Binary synthesis, FPGA, configurable logic, hardware/software codesign, hardware/software partitioning, synthesis from software binaries, warp processors

#### 19 VISTA: VPO interactive system for tuning applications



Prasad Kulkarni, Wankang Zhao, Stephen Hines, David Whalley, Xin Yuan, Robert van Engelen, Kyle Gallivan, Jason Hiser, Jack Davidson, Baosheng Cai, Mark Bailey, Hwashin Moon, Kyunghwan Cho, Yunheung Paek  
November 2006 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 5 Issue 4

**Publisher:** ACM Press

Full text available: [pdf\(4.01 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Software designers face many challenges when developing applications for embedded systems. One major challenge is meeting the conflicting constraints of speed, code size, and power consumption. Embedded application developers often resort to hand-coded assembly language to meet these constraints since traditional optimizing compiler technology is usually of little help in addressing this challenge. The results are software systems that are not portable, less robust, and more costly to develop an ...

**Keywords:** User-directed code improvement, genetic algorithms, interactive compilation, phase ordering

#### 20 A Code Transformation-Based Methodology for Improving I-Cache Performance of DSP Applications

N. Liveris, N. Zervas, D. Soudris, C. Goutis

March 2002 **Proceedings of the conference on Design, automation and test in Europe DATE '02**

**Publisher:** IEEE Computer Society

Full text available: [pdf\(105.95 KB\)](#) Additional Information: [full citation](#), [abstract](#), [citations](#)

This paper focuses on I-cache behaviour enhancement through the application of high-level code transformations. Specifically, a flow for the iterative application of the I-Cache performance optimizing transformations is proposed. The procedure of applying transformation is driven by a set of analytical equations, which receive parameters related to code and I-cache structure and predict the number of I-cache misses. Experimental results from a real-life demonstration application shows that order of magnit ...

Results 1 - 20 of 200

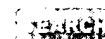
Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.  
[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)


[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

 Search: ☒ The ACM Digital Library ☐ The Guide


[Feedback](#) [Report a problem](#) [Satisfaction survey](#)
Terms used: **critical code header file**

Found 1,745 of 215,737

Sort results by

[Save results to a Binder](#)[Try an Advanced Search](#)

Display results

[Search Tips](#)[Try this search in The ACM Guide](#)
☐ Open results in a new window

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance scale ☐ ☐ ☐ ☐ ☐

# 1 [Fast and flexible application-level networking on exokernel systems](#)



Gregory R. Ganger, Dawson R. Engler, M. Frans Kaashoek, Hector M. Briceño, Russell Hunt, Thomas Pinckney

 February 2002 **ACM Transactions on Computer Systems (TOCS)**, Volume 20 Issue 1

Publisher: ACM Press

 Full text available: [pdf\(500.67 KB\)](#)

 Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Application-level networking is a promising software organization for improving performance and functionality for important network services. The Xok/ExOS exokernel system includes application-level support for standard network services, while at the same time allowing application writers to specialize networking services. This paper describes how Xok/ExOS's kernel mechanisms and library operating system organization achieve this flexibility, and retrospectively shares our experiences an ...

**Keywords:** Extensible systems, OS structure, fast servers, network services

# 2 [Compressed inverted files with reduced decoding overheads](#)



Anh Ngoc Vo, Alistair Moffat

 August 1998 **Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval SIGIR '98**

Publisher: ACM Press

 Full text available: [pdf\(1.09 MB\)](#)

 Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

# 3 [A C++ data model supporting reachability analysis and dead code detection](#)



Yih-Farn R. Chen, Emden R. Gansner, Eleftherios Koutsoufios

 November 1997 **ACM SIGSOFT Software Engineering Notes**, **Proceedings of the 6th European conference held jointly with the 5th ACM SIGSOFT international symposium on Foundations of software engineering ESEC '97/FSE-5**, Volume 22 Issue 6

Publisher: Springer-Verlag New York, Inc., ACM Press

 Full text available: [pdf\(1.33 MB\)](#)

 Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

# 4 [C-CLR: a tool for navigating highly configurable system software](#)



Nieraj Singh, Celina Gibbs, Yvonne Coady

 March 2007 **Proceedings of the 6th workshop on Aspects, components, and patterns for infrastructure software ACP4IS '07**



**Publisher:** ACM Press

Full text available:  pdf(249.17 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

In order to accommodate the spectrum of configuration options currently required for competitive system infrastructures, many systems leverage heavy usage of C preprocessor controlled conditional compilation. Inherent costs associated with this heavy preprocessor usage include both the impaired readability of the base system, and the reduced reusability of the configuration code.

Our proposed solution, C-CLR, allows developers to sift through views of a system based on configuration o ...

**Keywords:** aspect-oriented programming, conditional compilation, modularization, preprocessor directives, structured programming, system configuration, tools

## 5 Operating system: Server operating systems



M. Frans Kaashoek, Dawson R. Engler, Gregory R. Ganger, Deborah A. Wallach  
September 1996 **Proceedings of the 7th workshop on ACM SIGOPS European workshop: Systems support for worldwide applications EW 7**

**Publisher:** ACM Press

Full text available:  pdf(862.39 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)


We introduce server operating systems, which are sets of abstractions and runtime support for specialized, high-performance server applications. We have designed and are implementing a prototype server OS with support for aggressive specialization, direct device-to-device access, an event-driven organization, and dynamic compiler-assisted ILP. Using this server OS, we have constructed an HTTP server that outperforms servers running on a conventional OS by more than an order of magnitude and that ...

## 6 Workshop on Dynamic Analysis (WODA): Refactoring gcc using structure field access traces and concept analysis



Robert W. Bowdidge  
May 2005 **ACM SIGSOFT Software Engineering Notes , Proceedings of the third international workshop on Dynamic analysis WODA '05**, Volume 30 Issue 4

**Publisher:** ACM Press

Full text available:  pdf(221.88 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Refactoring usually involves statically analyzing source code to understand which transformations safely preserve execution behavior of the program. However, static analysis may not scale well for large programs when analysis results are too general, when tools for analyzing the source code are unwieldy, or when the tools simply do not exist. In such cases, it can be simpler to analyze the program at runtime to gather answers needed for safe code changes. I show how dynamic data can guide refactor ...

**Keywords:** case study, gcc, meaning-preserving restructuring

## 7 Collecting whole-system reference traces of multiprogrammed and multithreaded workloads



Scott F. Kaplan  
January 2004 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 4th international workshop on Software and performance WOSP '04**, Volume 29 Issue 1

**Publisher:** ACM Press

Full text available:  pdf(1.08 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

The simulated evaluation of memory management policies relies on *reference traces*--logs of memory operations performed by running processes. No existing approach to reference trace collection is applicable to a complete system, including the kernel and all processes. Specifically, none gather sufficient information for simulating the virtual memory

management, the filesystem cache management, and the scheduling of a multiprogrammed, multithreaded workload. Existing trace collectors are al ...

## 8 Distributed system V IPC in LOCUS: a design and implementation retrospective



B D Fleisch

August 1986 **ACM SIGCOMM Computer Communication Review , Proceedings of the ACM SIGCOMM conference on Communications architectures & protocols SIGCOMM '86**, Volume 16 Issue 3

**Publisher:** ACM Press

Full text available: [pdf\(1.30 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper describes new interprocess communications facilities that have been added to the Locus system [POPEK 81][WALKER 83]. The facilities improve Locus's interprocess communication repertoire by providing distributed support for three separate subsystems from System V UNIX: messages, semaphores, and shared memory. Here we describe these subsystems and their integration into in the Locus architecture.

## 9 Removing implementation details from C++ class declarations



Mark R. Headington

March 1995 **ACM SIGCSE Bulletin , Proceedings of the twenty-sixth SIGCSE technical symposium on Computer science education SIGCSE '95**, Volume 27 Issue 1

**Publisher:** ACM Press

Full text available: [pdf\(460.37 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Data abstraction—a concept introduced at varying places in the CS1/CS2/CS7 sequence—separates the properties of a data type (its values and operations) from the implementation of that type. This separation of specification from implementation is achieved by encapsulating the implementation so that users of the type can neither access nor be influenced by the implementation details. Ideally, therefore, the specification should be implementation-independent. The C++ clas ...

## 10 How do you release a product implemented in Ada?: making compatible interface changes with the BiiN Ada compiler



David B. Kinder

July 1989 **Proceedings of the sixth Washington Ada symposium on Ada WADAS '89**

**Publisher:** ACM Press

Full text available: [pdf\(670.48 KB\)](#)

Additional Information: [full citation](#), [references](#), [index terms](#)

## 11 Data base integrity and protection: Recovery and crash resistance in a filing system



Joost Verhofstad

August 1977 **Proceedings of the 1977 ACM SIGMOD international conference on Management of data SIGMOD '77**

**Publisher:** ACM Press

Full text available: [pdf\(1.13 MB\)](#)



Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

This paper describes mechanisms that provide the user of a filing system the dynamic facility for defining a scope within which backing out can be done on request. Check points (defining the beginning of a new scope) can dynamically be established and procedures for 'acceptance' (at the end of the scope) or 'undoing' (within or at the end of the scope) can be invoked. These scopes can be nested. It is also shown that these mechanisms can be used to provide crash resistance. After a crash the syste ...



**Keywords:** audit trail, backing out, consistency, crash resistance, error recovery, fault tolerance, filing system, recovery block, recovery cache (=recursive cache)

## 12



The Flux OSKit: a substrate for kernel and language research

-  Bryan Ford, Godmar Back, Greg Benson, Jay Lepreau, Albert Lin, Olin Shivers  
 October 1997 **ACM SIGOPS Operating Systems Review , Proceedings of the sixteenth ACM symposium on Operating systems principles SOSP '97**, Volume 31 Issue 5  
**Publisher:** ACM Press  
 Full text available:  [pdf\(2.47 MB\)](#) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

13 Standardisation—opportunity or constraint? (panel session)

-  David Arnold, Jack Bresenham, Ken Brodlie, George S. Carson, Jan Hardenbergh, Paul van Binst, Andries van Dam  
 September 1995 **Proceedings of the 22nd annual conference on Computer graphics and interactive techniques SIGGRAPH '95**  
**Publisher:** ACM Press  
 Full text available:  [pdf\(356.78 KB\)](#) Additional Information: [full citation](#), [references](#), [index terms](#)



14 IO-Lite: a unified I/O buffering and caching system

-  Vivek S. Pai, Peter Druschel, Willy Zwaenepoel  
 February 2000 **ACM Transactions on Computer Systems (TOCS)**, Volume 18 Issue 1  
**Publisher:** ACM Press  
 Full text available:  [pdf\(196.15 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This article presents the design, implementation, and evaluation of IO -Lite, a unified I/O buffering and caching system for general-purpose operating systems. IO-Lite unifies all buffering and caching in the system, to the extent permitted by the hardware. In particular, it allows applications, the interprocess communication system, the file system, the file cache, and the network subsystem to safely and concurrently share a single physical copy of the data. Protection and ...

**Keywords:** I/O buffering, caching, networking, zero-copy



15 Individualized exercises for self-assessment of programming knowledge: An evaluation of QuizPACK

-  Peter Brusilovsky, Sergey Sosnovsky  
 September 2005 **Journal on Educational Resources in Computing (JERIC)**, Volume 5 Issue 3  
**Publisher:** ACM Press  
 Full text available:  [pdf\(260.68 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Individualized exercises are a promising feature in promoting modern e-learning. The focus of this article is on the QuizPACK system, which is able to generate parameterized exercises for the C language and automatically evaluate the correctness of student answers. We introduce QuizPACK and present the results of its comprehensive classroom evaluation during four consecutive semesters. Our studies demonstrate that when QuizPACK is used for out-of-class self-assessment, it is an exceptional learn ...

**Keywords:** E-learning, assessment, classroom study, code execution, individualized exercises, introductory programming, parameterized questions

16 Computer education II: Computer tutoring for programming education

-  Susan M. Eitelman  
 March 2006 **Proceedings of the 44th annual Southeast regional conference ACM-SE 44**  
**Publisher:** ACM Press  
 Full text available:  [pdf\(158.22 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Software is increasingly pervasive in the products we use. Consequently, more programmers are needed to develop the software. However, there is also an unmet demand on programming instructors. One possible solution to the increased demand is to complement human teaching with automated computer tutoring. Several examples of such computer tutors for programming already exist, however they have not found widespread success. In the operational world, there are several job-aids that support programme ...

**Keywords:** intelligent tutoring

# 17 Explicit memory management: Safe manual memory management

David Gay, Rob Ennals, Eric Brewer

October 2007 **Proceedings of the 6th international symposium on Memory management ISMM '07**

**Publisher:** ACM

Full text available:  [pdf\(272.06 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

We present HeapSafe, a tool that uses reference counting to dynamically verify the soundness of manual memory management of C programs. HeapSafe relies on a simple extension to the usual malloc/free memory management API: delayed free scopes during which otherwise dangling references can exist. Porting programs for use with HeapSafe typically requires little effort (on average 0.6% of lines change), adds an average 11% time overhead (84% in the worst case), and increases space usage by an average ...

**Keywords:** C, memory management, reference counting, safety

# 18 Efficient, Unified, and Scalable Performance Monitoring for Multiprocessor Operating Systems

Robert W. Wisniewski, Bryan Rosenberg

November 2003 **Proceedings of the 2003 ACM/IEEE conference on Supercomputing SC '03**

**Publisher:** IEEE Computer Society

Full text available:  [pdf\(250.19 KB\)](#) Additional Information: [full citation](#), [abstract](#), [citations](#)

Programming, understanding, and tuning the performance of large multiprocessor systems is challenging. Experts have difficulty achieving good utilization for applications on large machines. The task of implementing a scalable system such as an operating system or database on large machines is even more challenging. And the importance of achieving good performance on multiprocessor machines is increasing as the number of cores per chip increases and as the size of multiprocessors increases. Cruci ...

# 19 How to port Linux when the hardware turns soft

David Lynch

January 2007 **Linux Journal**, Volume 2007 Issue 153

**Publisher:** Specialized Systems Consultants, Inc.

Full text available:  [html\(287.14 KB\)](#) Additional Information: [full citation](#), [abstract](#), [index terms](#)

Soul of the Pico machines


# 20 VISTA: VPO interactive system for tuning applications



Prasad Kulkarni, Wankang Zhao, Stephen Hines, David Whalley, Xin Yuan, Robert van Engelen, Kyle Gallivan, Jason Hiser, Jack Davidson, Baosheng Cai, Mark Bailey, Hwashin Moon, Kyunghwan Cho, Yunheung Paek

November 2006 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 5 Issue 4

**Publisher:** ACM Press

Full text available:  [pdf\(4.01 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Software designers face many challenges when developing applications for embedded

systems. One major challenge is meeting the conflicting constraints of speed, code size, and power consumption. Embedded application developers often resort to hand-coded assembly language to meet these constraints since traditional optimizing compiler technology is usually of little help in addressing this challenge. The results are software systems that are not portable, less robust, and more costly to develop an ...



**Keywords:** User-directed code improvement, genetic algorithms, interactive compilation, phase ordering

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)


☐ Search Results

## BROWSE

## SEARCH

## IEEE XPLORE GUIDE

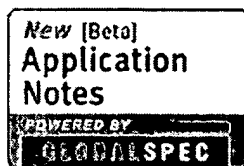
## SUPPORT

Results for "((critical and code and compil\*)&lt;in&gt;metadata)"

Your search matched 168 of 1692897 documents.

A maximum of 100 results are displayed, 25 to a page, sorted by Relevance in Descending order.

e-mail
 printer



## Modify Search


☐ Check to search only within this results set

 Display Format: ☒ Citation ☐ Citation & Abstract

## » Search Options

[View Session History](#)
[New Search](#)

## » Key

IEEE JNL	IEEE Journal or Magazine
IET JNL	IET Journal or Magazine
IEEE CNF	IEEE Conference Proceeding
IET CNF	IET Conference Proceeding
IEEE STD	IEEE Standard

## IEEE/IET

## Books

## Educational Courses

## Application Notes [

IEEE/IET journals, transactions, letters, magazines, conference proceedings, and standards.

[Select All](#) [Deselect All](#)
View: [1-25](#) | [26-50](#) | [51-75](#)

- ☐ 1. **New Life in Dusty Decks: Results of Porting a CM Fortran Based Aeroacoustic Model to Hi Performance Fortran**  
 Nucciarone, J.J.; Ozyoruk, Y.; Long, L.N.;  
[Supercomputing, ACM/IEEE 1997 Conference](#)  
 15-21 Nov. 1997 Page(s):16 - 16  
 Digital Object Identifier 10.1109/SC.1997.10044  
[AbstractPlus](#) | Full Text: [PDF](#)(184 KB) IEEE CNF  
[Rights and Permissions](#)
- ☐ 2. **An efficient compiler technique for code size reduction using reduced bit-width ISAs**  
 Halambi, A.; Shrivastava, A.; Biswas, P.; Dutt, N.; Nicolau, A.;  
[Design, Automation and Test in Europe Conference and Exhibition, 2002. Proceedings](#)  
 4-8 March 2002 Page(s):402 - 408  
 Digital Object Identifier 10.1109/DATE.2002.998305  
[AbstractPlus](#) | Full Text: [PDF](#)(419 KB) IEEE CNF  
[Rights and Permissions](#)
- ☐ 3. **A practitioner report on the evaluation of the performance of the C, C++ and Java compiler the OS/390 platform**  
 Cargill, D.A.; Radaideh, M.;  
[Performance Analysis of Systems and Software, 2000. ISPASS, 2000 IEEE International Sympos](#)  
 24-25 April 2000 Page(s):40 - 45  
 Digital Object Identifier 10.1109/ISPASS.2000.842279  
[AbstractPlus](#) | Full Text: [PDF](#)(240 KB) IEEE CNF  
[Rights and Permissions](#)
- ☐ 4. **A Code Refinement Methodology for Performance-Improved Synthesis from C**  
 Stitt, G.; Vahid, F.; Najjar, W.;  
[Computer-Aided Design, 2006. ICCAD '06. IEEE/ACM International Conference on](#)  
 Nov. 2006 Page(s):716 - 723  
 Digital Object Identifier 10.1109/ICCAD.2006.320018  
[AbstractPlus](#) | Full Text: [PDF](#)(10517 KB) IEEE CNF  
[Rights and Permissions](#)
- ☐ 5. **Optimizing address code generation for array-intensive DSP applications**  
 Guilin Chen; Kandemir, M.;  
[Code Generation and Optimization, 2005. CGO 2005. International Symposium on](#)  
 20-23 March 2005 Page(s):141 - 152

Digital Object Identifier 10.1109/CGO.2005.23

[AbstractPlus](#) | [Full Text: PDF\(520 KB\)](#) IEEE CNF

[Rights and Permissions](#)

6. **Countering trusting trust through diverse double-compiling**  
Wheeler, D.A.;  
[Computer Security Applications Conference, 21st Annual](#)  
5-9 Dec. 2005 Page(s):13 pp.  
Digital Object Identifier 10.1109/CSAC.2005.17  
[AbstractPlus](#) | [Full Text: PDF\(280 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
7. **An efficient algorithm for pointer-to-array access conversion for compiling and optimizing applications**  
van Engelen, R.A.; Gallivan, K.A.;  
[Innovative Architecture for Future Generation High-Performance Processors and Systems, 2001](#)  
18-19 Jan. 2001 Page(s):80 - 89  
Digital Object Identifier 10.1109/IWIA.2001.955200  
[AbstractPlus](#) | [Full Text: PDF\(712 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
8. **Compiling for EPIC architectures**  
Kathail, V.; Schlansker, M.S.; Rau, B.R.;  
[Proceedings of the IEEE](#)  
Volume 89, Issue 11, Nov. 2001 Page(s):1676 - 1693  
Digital Object Identifier 10.1109/5.964445  
[AbstractPlus](#) | [References](#) | [Full Text: PDF\(197 KB\)](#) | [Full Text: HTML](#) IEEE JNL  
[Rights and Permissions](#)
9. **On the requirements of high-integrity code generation**  
Whalen, M.W.; Heimdahl, M.P.E.;  
[High-Assurance Systems Engineering, 1999. Proceedings. 4th IEEE International Symposium on](#)  
17-19 Nov. 1999 Page(s):217 - 224  
Digital Object Identifier 10.1109/HASE.1999.809497  
[AbstractPlus](#) | [Full Text: PDF\(84 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
10. **Can safety-critical software be flexible?**  
Fraser, S.W.;  
[Information Reuse and Integration, 2003. IRI 2003. IEEE International Conference on](#)  
2003 Page(s):588 - 593  
Digital Object Identifier 10.1109/IRI.2003.1251470  
[AbstractPlus](#) | [Full Text: PDF\(422 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
11. **Optimal code size reduction for software-pipelined loops on DSP applications**  
Qingfeng Zhuge; Zili Shao; Sha, E.H.-M.;  
[Parallel Processing, 2002. Proceedings. International Conference on](#)  
18-21 Aug. 2002 Page(s):613 - 620  
Digital Object Identifier 10.1109/ICPP.2002.1040919  
[AbstractPlus](#) | [Full Text: PDF\(330 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
12. **Using global code motions to improve the quality of results for high-level synthesis**  
Gupta, S.; Savoiu, N.; Dutt, N.; Gupta, R.; Nicolau, A.;  
[Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on](#)  
Volume 23, Issue 2, Feb. 2004 Page(s):302 - 312  
Digital Object Identifier 10.1109/TCAD.2003.822105  
[AbstractPlus](#) | [References](#) | [Full Text: PDF\(960 KB\)](#) IEEE JNL  
[Rights and Permissions](#)
13. **Compiler and runtime support for running OpenMP programs on Pentium- and Itanium- architectures**

Xinmin Tian; Girkar, M.; Shah, S.; Armstrong, D.; Su, E.; Petersen, P.;  
High-Level Parallel Programming Models and Supportive Environments, 2003. Proceedings. Eighth International Workshop on  
22 April 2003 Page(s):47 - 55  
[AbstractPlus](#) | Full Text: [PDF\(297 KB\)](#) IEEE CNF  
[Rights and Permissions](#)

**14. Do you trust your compiler?**

Boyle, J.M.; Resler, R.D.; Winter, V.L.;  
Computer  
Volume 32, Issue 5, May 1999 Page(s):65 - 73  
Digital Object Identifier 10.1109/2.762804  
[AbstractPlus](#) | [References](#) | Full Text: [PDF\(144 KB\)](#) IEEE JNL  
[Rights and Permissions](#)

**15. VHC: quickly building an optimizer for complex embedded architectures**

Dupre, M.; Drach, N.; Temam, O.;  
Code Generation and Optimization, 2004. CGO 2004. International Symposium on  
2004 Page(s):53 - 64  
Digital Object Identifier 10.1109/CGO.2004.1281663  
[AbstractPlus](#) | Full Text: [PDF\(1322 KB\)](#) IEEE CNF  
[Rights and Permissions](#)

**16. Compiler and runtime support for running OpenMP programs on Pentium- and Itanium- architectures**

Xinmin Tian; Girkar, M.; Shah, S.; Armstrong, D.; Su, E.; Petersen, P.;  
Parallel and Distributed Processing Symposium, 2003. Proceedings. International  
22-26 April 2003 Page(s):9 pp.  
Digital Object Identifier 10.1109/IPDPS.2003.1213252  
[AbstractPlus](#) | Full Text: [PDF\(298 KB\)](#) IEEE CNF  
[Rights and Permissions](#)

**17. An efficient VLSI architecture of VLD for AVS HDTV decoder**

Bin Sheng; Wen Gao; Don Xie; Di Wu;  
Consumer Electronics, IEEE Transactions on  
Volume 52, Issue 2, May 2006 Page(s):696 - 701  
Digital Object Identifier 10.1109/TCE.2006.1649699  
[AbstractPlus](#) | Full Text: [PDF\(220 KB\)](#) IEEE JNL  
[Rights and Permissions](#)

**18. Building intrusion-tolerant secure software**

Zhang, T.; Zhuang, X.; Pande, S.;  
Code Generation and Optimization, 2005. CGO 2005. International Symposium on  
20-23 March 2005 Page(s):255 - 266  
Digital Object Identifier 10.1109/CGO.2005.8  
[AbstractPlus](#) | Full Text: [PDF\(240 KB\)](#) IEEE CNF  
[Rights and Permissions](#)

**19. ADAPT: Automated De-coupled Adaptive Program Transformation**

Voss, M.J.; Eigenmann, R.;  
Parallel Processing, 2000. Proceedings. 2000 International Conference on  
21-24 Aug. 2000 Page(s):163 - 170  
Digital Object Identifier 10.1109/ICPP.2000.876107  
[AbstractPlus](#) | Full Text: [PDF\(680 KB\)](#) IEEE CNF  
[Rights and Permissions](#)

**20. Portable code for complex critical systems**

Audsley, N.C.; Bate, I.J.; Grigg, A.;  
Real-Time Computing Systems and Applications, 1999. RTCSA '99. Sixth International Conferen  
13-15 Dec. 1999 Page(s):111 - 118  
Digital Object Identifier 10.1109/RTCSA.1999.811200  
[AbstractPlus](#) | Full Text: [PDF\(692 KB\)](#) IEEE CNF  
[Rights and Permissions](#)



21. **Practical instruction set design and compiler retargetability using static resource models**  
Qin Zhao; Mesman, B.; Basten, T.;  
Design, Automation and Test in Europe Conference and Exhibition, 2002. Proceedings  
4-8 March 2002 Page(s):1021 - 1026  
Digital Object Identifier 10.1109/DATE.2002.998425  
[AbstractPlus](#) | Full Text: [PDF](#)(259 KB) [IEEE CNF](#)  
[Rights and Permissions](#)
22. **Compilation scheme for near fine grain parallel processing on a multiprocessor system with explicit synchronization**  
Ogata, W.; Fujimoto, K.; Oota, M.; Kasahara, H.;  
Communications, Computers, and Signal Processing, 1995. Proceedings. IEEE Pacific Rim Conference  
17-19 May 1995 Page(s):327 - 332  
Digital Object Identifier 10.1109/PACRIM.1995.519536  
[AbstractPlus](#) | Full Text: [PDF](#)(472 KB) [IEEE CNF](#)  
[Rights and Permissions](#)
23. **Models for automatic generation of safety-critical real-time systems**  
Buckl, Christian; Regensburger, Matthias; Knoll, Alois; Schrott, Gerhard;  
Availability, Reliability and Security, 2007. ARES 2007. The Second International Conference on  
10-13 April 2007 Page(s):580 - 587  
Digital Object Identifier 10.1109/ARES.2007.106  
[AbstractPlus](#) | Full Text: [PDF](#)(353 KB) [IEEE CNF](#)  
[Rights and Permissions](#)
24. **Model-based development for time-triggered architectures**  
Dion, B.; Le Sergent, T.; Martin, B.; Griebel, H.;  
Digital Avionics Systems Conference, 2004. DASC 04. The 23rd  
Volume 2, 24-28 Oct. 2004 Page(s):6.D.3 - 6.1-7 Vol.2  
[AbstractPlus](#) | Full Text: [PDF](#)(1011 KB) [IEEE CNF](#)  
[Rights and Permissions](#)
25. **Optimal code size reduction for software-pipelined and unfolded loops**  
Qingfeng Zhuge; Bin Xiao; Zili Shao; Sha, E.H.-M.; Chantana Chantrapornchai;  
System Synthesis, 2002. 15th International Symposium on  
2002 Page(s):144 - 149  
[AbstractPlus](#) | Full Text: [PDF](#)(520 KB) [IEEE CNF](#)  
[Rights and Permissions](#)

View: [1-25](#) | [26-50](#) | [51-75](#)

[Help](#) [Contact Us](#) [Privacy & Security](#) |

© Copyright 2007 IEEE – All Rights Reserved


☐ Search Results

## BROWSE

## SEARCH

## IEEE XPLORE GUIDE

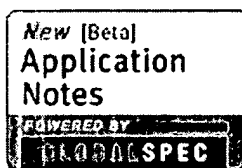
## SUPPORT

Results for "((critical and code and optimiz\*)&lt;in&gt;metadata)"

Your search matched 262 of 1692897 documents.

A maximum of 100 results are displayed, 25 to a page, sorted by Relevance in Descending order.

e-mail
 printer



## Modify Search


☐ Check to search only within this results set

 Display Format: ☒ Citation ☐ Citation & Abstract

## » Search Options

[View Session History](#)
[New Search](#)

## » Key

IEEE JNL IEEE Journal or Magazine

IET JNL IET Journal or Magazine

IEEE CNF IEEE Conference Proceeding

IET CNF IET Conference Proceeding

IEEE STD IEEE Standard

## IEEE/IET

## Books

## Educational Courses

## Application Notes [

IEEE/IET journals, transactions, letters, magazines, conference proceedings, and standards.

 
View: 1-25 | [26-50](#) | [51-75](#)

- ☐ 1. **Optimizing address code generation for array-intensive DSP applications**  
 Guilin Chen; Kandemir, M.;  
[Code Generation and Optimization, 2005. CGO 2005. International Symposium on 20-23 March 2005 Page\(s\):141 - 152](#)  
 Digital Object Identifier 10.1109/CGO.2005.23  
[AbstractPlus](#) | Full Text: [PDF\(520 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
- ☐ 2. **Performance of runtime optimization on BLAST**  
 Das, A.; Jiwei Lu; Chen, H.; Kim, J.; Pen-Chung Yew; Wei-Chung Hsu; Dong-Yuan Chen;  
[Code Generation and Optimization, 2005. CGO 2005. International Symposium on 20-23 March 2005 Page\(s\):86 - 96](#)  
 Digital Object Identifier 10.1109/CGO.2005.25  
[AbstractPlus](#) | Full Text: [PDF\(224 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
- ☐ 3. **VHC: quickly building an optimizer for complex embedded architectures**  
 Dupre, M.; Drach, N.; Temam, O.;  
[Code Generation and Optimization, 2004. CGO 2004. International Symposium on 2004 Page\(s\):53 - 64](#)  
 Digital Object Identifier 10.1109/CGO.2004.1281663  
[AbstractPlus](#) | Full Text: [PDF\(1322 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
- ☐ 4. **Source code optimization and profiling of energy consumption in embedded systems**  
 Simunic, T.; Benini, L.; De Micheli, G.; Hans, M.;  
[System Synthesis, 2000. Proceedings. The 13th International Symposium on 20-22 Sept. 2000 Page\(s\):193 - 198](#)  
 Digital Object Identifier 10.1109/ISSS.2000.874049  
[AbstractPlus](#) | Full Text: [PDF\(528 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
- ☐ 5. **On the robustness of vector set partitioning image coders to bit errors**  
 Mukherjee, D.; Mitra, S.K.;  
[Circuits and Systems, 1999. ISCAS '99. Proceedings of the 1999 IEEE International Symposium Volume 4, 30 May-2 June 1999 Page\(s\):41 - 45 vol.4](#)  
 Digital Object Identifier 10.1109/ISCAS.1999.779938

[AbstractPlus](#) | Full Text: [PDF\(544 KB\)](#) IEEE CNF  
[Rights and Permissions](#)

6. **A vector set partitioning noisy channel image coder with unequal error protection**  
Mukherjee, D.; Mitra, S.K.;  
[Selected Areas in Communications, IEEE Journal on](#)  
Volume 18, [Issue 6](#), June 2000 Page(s):829 - 840  
Digital Object Identifier 10.1109/49.848237  
[AbstractPlus](#) | [References](#) | Full Text: [PDF\(232 KB\)](#) IEEE JNL  
[Rights and Permissions](#)
7. **An efficient algorithm for pointer-to-array access conversion for compiling and optimizing applications**  
van Engelen, R.A.; Gallivan, K.A.;  
[Innovative Architecture for Future Generation High-Performance Processors and Systems, 2001](#)  
18-19 Jan. 2001 Page(s):80 - 89  
Digital Object Identifier 10.1109/IWIA.2001.955200  
[AbstractPlus](#) | Full Text: [PDF\(712 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
8. **Code Combining--A Maximum-Likelihood Decoding Approach for Combining an Arbitrary Number of Noisy Packets**  
Chase, D.;  
[Communications, IEEE Transactions on \[legacy, pre - 1988\]](#)  
Volume 33, [Issue 5](#), May 1985 Page(s):385 - 393  
[AbstractPlus](#) | Full Text: [PDF\(880 KB\)](#) IEEE JNL  
[Rights and Permissions](#)
9. **Computational Techniques for Pulsed Power Design**  
Spielman, R.B.;  
[Power Modulator Symposium, 2006. Conference Record of the 2006 Twenty-Seventh International](#)  
14-18 May 2006 Page(s):43 - 48  
Digital Object Identifier 10.1109/MODSYM.2006.365179  
[AbstractPlus](#) | Full Text: [PDF\(716 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
10. **The Embedded Genetic Allocator-a system to automatically optimize the use of memory resources in high performance, scalable computing systems**  
Cousins, D.; Loomis, J.; Roeber, F.; Schoeppner, P.; Tobin, A.-E.;  
[Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on](#)  
Volume 3, 11-14 Oct. 1998 Page(s):2166 - 2171 vol.3  
Digital Object Identifier 10.1109/ICSMC.1998.724976  
[AbstractPlus](#) | Full Text: [PDF\(668 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
11. **A concatenated two-stage adaptive (CTSA) error control scheme for data transmission in fading channels**  
Cygan, D.; Lutz, E.;  
[Communications, IEEE Transactions on](#)  
Volume 43, [Issue 234](#), Feb-Mar-Apr 1995 Page(s):795 - 803  
Digital Object Identifier 10.1109/26.380111  
[AbstractPlus](#) | Full Text: [PDF\(872 KB\)](#) IEEE JNL  
[Rights and Permissions](#)
12. **Analysis of coded noncoherent transmission in DS-SS mobile satellite communication**  
Corazza, G.E.; De Gaudenzi, R.;  
[Communications, IEEE Transactions on](#)  
Volume 46, [Issue 11](#), Nov. 1998 Page(s):1525 - 1535  
Digital Object Identifier 10.1109/26.729397  
[AbstractPlus](#) | [References](#) | Full Text: [PDF\(448 KB\)](#) IEEE JNL  
[Rights and Permissions](#)
13. **Efficient audio coding with optimized subband configurations**

Ali, M.T.; Mian, M.S.;  
Applications of Signal Processing to Audio and Acoustics, 2005. IEEE Workshop on  
16-19 Oct. 2005 Page(s):223 - 226  
Digital Object Identifier 10.1109/ASPAA.2005.1540210  
[AbstractPlus](#) | [Full Text: PDF\(165 KB\)](#) IEEE CNF  
[Rights and Permissions](#)

14. **Fast, optimized Sun RPC using automatic program specialization**  
Muller, G.; Marlet, R.; Volanschi, E.-N.; Consel, C.; Pu, C.; Goel, A.;  
Distributed Computing Systems, 1998. Proceedings. 18th International Conference on  
26-29 May 1998 Page(s):240 - 249  
Digital Object Identifier 10.1109/ICDCS.1998.679507  
[AbstractPlus](#) | [Full Text: PDF\(100 KB\)](#) IEEE CNF  
[Rights and Permissions](#)

15. **Querying source code using an algebraic query language**  
Paul, S.; Prakash, A.;  
Software Maintenance, 1994. Proceedings., International Conference on  
19-23 Sept. 1994 Page(s):127 - 136  
Digital Object Identifier 10.1109/ICSM.1994.336782  
[AbstractPlus](#) | [Full Text: PDF\(672 KB\)](#) IEEE CNF  
[Rights and Permissions](#)

16. **ADAPT: Automated De-coupled Adaptive Program Transformation**  
Voss, M.J.; Eigenmann, R.;  
Parallel Processing, 2000. Proceedings. 2000 International Conference on  
21-24 Aug. 2000 Page(s):163 - 170  
Digital Object Identifier 10.1109/ICPP.2000.876107  
[AbstractPlus](#) | [Full Text: PDF\(680 KB\)](#) IEEE CNF  
[Rights and Permissions](#)

17. **Optimal context quantization in lossless compression of image data sequences**  
Forchhammer, S.; Xiaolin Wu; Andersen, J.D.;  
Image Processing, IEEE Transactions on  
Volume 13, Issue 4, April 2004 Page(s):509 - 517  
Digital Object Identifier 10.1109/TIP.2003.822613  
[AbstractPlus](#) | [References](#) | [Full Text: PDF\(336 KB\)](#) IEEE JNL  
[Rights and Permissions](#)

18. **Enhanced Features for Design of Traveling Wave Tubes Using CHRISTINE-1D**  
David, J.A.; Kory, C.L.; Tran, H.T.; Ives, R.L.; Chernin, D.;  
Plasma Science, IEEE Transactions on  
Volume 35, Issue 4, Part 3, Aug. 2007 Page(s):1056 - 1064  
Digital Object Identifier 10.1109/TPS.2007.902128  
[AbstractPlus](#) | [Full Text: PDF\(298 KB\)](#) IEEE JNL  
[Rights and Permissions](#)

19. **SPIRAL: code generation for DSP transforms**  
Puschel, M.; Moura, J.M.F.; Johnson, J.R.; Padua, D.; Veloso, M.M.; Singer, B.W.; Jianxin Xiong  
Franchetti, F.; Gacic, A.; Voronenko, Y.; Chen, K.; Johnson, R.W.; Rizzolo, N.;  
Proceedings of the IEEE  
Volume 93, Issue 2, Feb 2005 Page(s):232 - 275  
Digital Object Identifier 10.1109/JPROC.2004.840306  
[AbstractPlus](#) | [Full Text: PDF\(1896 KB\)](#) IEEE JNL  
[Rights and Permissions](#)

20. **Analysis and architecture design of block-coding engine for EBCOT in JPEG 2000**  
Chung-Jr Lian; Kuan-Fu Chen; Hong-Hui Chen; Liang-Gee Chen;  
Circuits and Systems for Video Technology, IEEE Transactions on  
Volume 13, Issue 3, March 2003 Page(s):219 - 230  
Digital Object Identifier 10.1109/TCSVT.2003.809833  
[AbstractPlus](#) | [References](#) | [Full Text: PDF\(1431 KB\)](#) IEEE JNL  
[Rights and Permissions](#)

21. **Optimal code size reduction for software-pipelined and unfolded loops**  
Qingfeng Zhuge; Bin Xiao; Zili Shao; Sha, E.H.-M.; Chantana Chantrapornchai;  
System Synthesis, 2002. 15th International Symposium on  
2002 Page(s):144 - 149  
[AbstractPlus](#) | Full Text: [PDF\(520 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
22. **A high-performance architecture for embedded block coding in JPEG 2000**  
Pastuszak, G.;  
Circuits and Systems for Video Technology, IEEE Transactions on  
Volume 15, Issue 9, Sept. 2005 Page(s):1182 - 1191  
Digital Object Identifier 10.1109/TCSVT.2005.852720  
[AbstractPlus](#) | Full Text: [PDF\(496 KB\)](#) IEEE JNL  
[Rights and Permissions](#)
23. **RD Optimal Temporal Noise Shaping for Transform Audio Coding**  
Niamut, O.A.; Heusdens, R.;  
Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on  
Volume 5, 14-19 May 2006 Page(s):V - V  
Digital Object Identifier 10.1109/ICASSP.2006.1661244  
[AbstractPlus](#) | Full Text: [PDF\(424 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
24. **Synthesis filters design for coding gain in oversampled filter banks**  
Labeau, F.;  
Signal Processing Letters, IEEE  
Volume 12, Issue 10, Oct. 2005 Page(s):697 - 700  
Digital Object Identifier 10.1109/LSP.2005.855549  
[AbstractPlus](#) | Full Text: [PDF\(224 KB\)](#) IEEE JNL  
[Rights and Permissions](#)
25. **A scalable loop optimization approach for scalable DSP processors**  
Jian Wang; Bogong Su; Erh-Wen Hu;  
Acoustics, Speech, and Signal Processing, 2000. ICASSP '00. Proceedings. 2000 IEEE International Conference on  
Volume 6, 5-9 June 2000 Page(s):3646 - 3649 vol.6  
Digital Object Identifier 10.1109/ICASSP.2000.860192  
[AbstractPlus](#) | Full Text: [PDF\(284 KB\)](#) IEEE CNF  
[Rights and Permissions](#)

View: 1-25 | 26-50 | 51-75

[Help](#) [Contact Us](#) [Privacy & Security](#)

© Copyright 2007 IEEE – All Rights Reserved



Welcome United States Patent and Trademark Office

☐ Search Results

BROWSE

SEARCH

IEEE XPLORE GUIDE

SUPPORT

Results for "((nest\* and loop\* and optimiz\*)&lt;in&gt;metadata)"

Your search matched 138 of 1692897 documents.

A maximum of 100 results are displayed, 25 to a page, sorted by Relevance in Descending order.

e-mail
 printer



Modify Search

Search &gt;

☐ Check to search only within this results set
Display Format: ☒ Citation ☐ Citation & Abstract

» Search Options

[View Session History](#)[New Search](#)

IEEE/IET

Books

Educational Courses

Application Notes [

IEEE/IET journals, transactions, letters, magazines, conference proceedings, and standards.

» Key

[view selected items](#)[Select All](#) [Deselect All](#)View: [1-25](#) | [26-50](#) | [51-75](#)

IEEE JNL IEEE Journal or Magazine

IET JNL IET Journal or Magazine

IEEE CNF IEEE Conference Proceeding

IET CNF IET Conference Proceeding

IEEE STD IEEE Standard

- ☐ 1. **A matrix-based approach to the global locality optimization problem**  
 Kandemir, M.; Choudhary, A.; Ramanujam, J.; Banerjee, P.;  
[Parallel Architectures and Compilation Techniques, 1998. Proceedings. 1998 International Conference on](#)  
 12-18 Oct. 1998 Page(s):306 - 313  
 Digital Object Identifier 10.1109/PACT.1998.727266  
[AbstractPlus](#) | Full Text: [PDF](#)(188 KB) IEEE CNF  
[Rights and Permissions](#)
- ☐ 2. **Loop coalescing and scheduling for barrier MIMD architectures**  
 O'Keefe, M.T.; Dietz, H.G.;  
[Parallel and Distributed Systems, IEEE Transactions on](#)  
 Volume 4, [Issue 9](#), Sept. 1993 Page(s):1060 - 1064  
 Digital Object Identifier 10.1109/71.243531  
[AbstractPlus](#) | Full Text: [PDF](#)(396 KB) IEEE JNL  
[Rights and Permissions](#)
- ☐ 3. **Quasidynamic layout optimizations for improving data locality**  
 Kadayif, I.; Kandemir, M.;  
[Parallel and Distributed Systems, IEEE Transactions on](#)  
 Volume 15, [Issue 11](#), Nov. 2004 Page(s):996 - 1011  
 Digital Object Identifier 10.1109/TPDS.2004.70  
[AbstractPlus](#) | [References](#) | Full Text: [PDF](#)(1272 KB) IEEE JNL  
[Rights and Permissions](#)
- ☐ 4. **Loop Nest Splitting for WCET-Optimization and Predictability Improvement**  
 Falk, H.; Schwarzer, M.;  
[Embedded Systems for Real Time Multimedia, Proceedings of the 2006 IEEE/ACM/IFIP Workshop](#)  
 Oct. 2006 Page(s):115 - 120  
 Digital Object Identifier 10.1109/ESTMED.2006.321283  
[AbstractPlus](#) | Full Text: [PDF](#)(232 KB) IEEE CNF  
[Rights and Permissions](#)
- ☐ 5. **Optimizing array-intensive applications for on-chip multiprocessors**  
 Kadayif, I.; Kandemir, M.; Chen, G.; Ozturk, O.; Karakoy, M.; Sezer, U.;  
[Parallel and Distributed Systems, IEEE Transactions on](#)  
 Volume 16, [Issue 5](#), May 2005 Page(s):396 - 411  
 Digital Object Identifier 10.1109/TPDS.2005.57

[AbstractPlus](#) | Full Text: [PDF\(1288 KB\)](#) IEEE JNL  
[Rights and Permissions](#)

6. **Optimizing Inter-Nest Data Locality Using Loop Splitting and Reordering**  
Naci, S.;  
[Parallel and Distributed Processing Symposium, 2007. IPDPS 2007. IEEE International](#)  
26-30 March 2007 Page(s):1 - 8  
Digital Object Identifier 10.1109/IPDPS.2007.370399  
[AbstractPlus](#) | Full Text: [PDF\(365 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
7. **Single-dimension software pipelining for multi-dimensional loops**  
Rong, H.; Zhizhong Tang; Govindarajan, R.; Douillet, A.; Gao, G.R.;  
[Code Generation and Optimization, 2004. CGO 2004. International Symposium on](#)  
2004 Page(s):163 - 174  
Digital Object Identifier 10.1109/CGO.2004.1281672  
[AbstractPlus](#) | Full Text: [PDF\(549 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
8. **Memory minimization for tensor contractions using integer linear programming**  
Allam, A.; Ramanujam, J.; Baumgartner, G.; Sadayappan, P.;  
[Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International](#)  
25-29 April 2006 Page(s):8 pp.  
Digital Object Identifier 10.1109/IPDPS.2006.1639717  
[AbstractPlus](#) | Full Text: [PDF\(296 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
9. **Timing optimization of nested loops considering code size for DSP applications**  
Qingfeng Zhuge; Zili Shao; Sha, E.H.-M.;  
[Parallel Processing, 2004. ICPP 2004. International Conference on](#)  
2004 Page(s):475 - 482 vol.1  
Digital Object Identifier 10.1109/ICPP.2004.1327957  
[AbstractPlus](#) | Full Text: [PDF\(396 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
10. **An optimized dependence convex hull partitioning technique to maximize parallelism of nested loops with non-uniform dependences**  
Der-Lin Pean; Guan-Joe Lai; Cheng Chen;  
[Parallel and Distributed Systems, 2000. Proceedings. Seventh International Conference on](#)  
4-7 July 2000 Page(s):367 - 374  
Digital Object Identifier 10.1109/ICPADS.2000.857719  
[AbstractPlus](#) | Full Text: [PDF\(492 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
11. **Schedule-driven loop unrolling for parallel processors**  
El-Rewini, H.; Lewis, T.;  
[System Sciences, 1991. Proceedings of the Twenty-Fourth Annual Hawaii International Conference on](#)  
Volume ii, 8-11 Jan. 1991 Page(s):458 - 467 vol.2  
Digital Object Identifier 10.1109/HICSS.1991.184008  
[AbstractPlus](#) | Full Text: [PDF\(668 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
12. **Control mechanism for software pipelining on nested loop**  
Tao Yu; Zhizhong Tang; Chihong Zhang; Jun Luo;  
[Advances in Parallel and Distributed Computing, 1997. Proceedings](#)  
19-21 March 1997 Page(s):345 - 350  
Digital Object Identifier 10.1109/APDC.1997.574053  
[AbstractPlus](#) | Full Text: [PDF\(624 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
13. **On the parallel execution time of tiled loops**  
Hogstedt, K.; Carter, L.; Ferrante, J.;  
[Parallel and Distributed Systems, IEEE Transactions on](#)

Volume 14, Issue 3, March 2003 Page(s):307 - 321

Digital Object Identifier 10.1109/TPDS.2003.1189587

[AbstractPlus](#) | [References](#) | Full Text: [PDF](#)(2219 KB) IEEE JNL

[Rights and Permissions](#)

14. **Integrating loop and data optimizations for locality within a constraint network based fram**  
Guilin Chen; Ozturk, O.; Kandemir, M.; Kolcu, I.;  
[Computer-Aided Design, 2005. ICCAD-2005. IEEE/ACM International Conference on](#)  
6-10 Nov. 2005 Page(s):279 - 282  
Digital Object Identifier 10.1109/ICCAD.2005.1560078  
[AbstractPlus](#) | Full Text: [PDF](#)(229 KB) IEEE CNF  
[Rights and Permissions](#)
15. **On the performance of hand vs. automatically optimized numerical codes**  
Jimenez, M.; Llaberia, J.M.; Fernandez, A.;  
[High-Performance Computer Architecture, 2000. HPCA-6. Proceedings. Sixth International Sympo](#)  
[on](#)  
8-12 Jan. 2000 Page(s):183 - 194  
Digital Object Identifier 10.1109/HPCA.2000.824349  
[AbstractPlus](#) | Full Text: [PDF](#)(384 KB) IEEE CNF  
[Rights and Permissions](#)
16. **Transformations for improving data access locality in non-perfectly nested loops**  
Kulkarni, D.;  
[Parallel Architectures and Compilation Techniques, 1998. Proceedings. 1998 International Confe](#)  
[on](#)  
12-18 Oct. 1998 Page(s):314 - 321  
Digital Object Identifier 10.1109/PACT.1998.727267  
[AbstractPlus](#) | Full Text: [PDF](#)(84 KB) IEEE CNF  
[Rights and Permissions](#)
17. **Hybrid loop interchange: optimization for parallel programs**  
Jin Guohua; Chen Fujie;  
[Parallel Processing Symposium, 1992. Proceedings., Sixth International](#)  
23-26 March 1992 Page(s):680 - 685  
Digital Object Identifier 10.1109/IPPS.1992.222984  
[AbstractPlus](#) | Full Text: [PDF](#)(368 KB) IEEE CNF  
[Rights and Permissions](#)
18. **An analytical model for loop tiling and its solution**  
Sarkar, V.; Megiddo, N.;  
[Performance Analysis of Systems and Software, 2000. ISPASS. 2000 IEEE International Sympo](#)  
24-25 April 2000 Page(s):146 - 153  
Digital Object Identifier 10.1109/ISPASS.2000.842294  
[AbstractPlus](#) | Full Text: [PDF](#)(300 KB) IEEE CNF  
[Rights and Permissions](#)
19. **Static scheduling of uniform nested loops**  
Chao, L.-F.; Sha, E.H.-M.;  
[Parallel Processing Symposium, 1993., Proceedings of Seventh International](#)  
13-16 April 1993 Page(s):254 - 258  
Digital Object Identifier 10.1109/IPPS.1993.262890  
[AbstractPlus](#) | Full Text: [PDF](#)(396 KB) IEEE CNF  
[Rights and Permissions](#)
20. **Improving data locality by array contraction**  
Yonghong Song; Rong Xu; Cheng Wang; Zhiyuan Li;  
[Computers, IEEE Transactions on](#)  
Volume 53, Issue 9, Sept. 2004 Page(s):1073 - 1084  
Digital Object Identifier 10.1109/TC.2004.62  
[AbstractPlus](#) | [References](#) | Full Text: [PDF](#)(1096 KB) IEEE JNL  
[Rights and Permissions](#)



21. **A unified framework for optimizing locality, parallelism, and communication in out-of-core computations**  
Kandemir, M.; Choudhary, A.; Ramanujam, J.; Kandaswamy, M.A.;  
Parallel and Distributed Systems, IEEE Transactions on  
Volume 11, Issue 7, July 2000 Page(s):648 - 668  
Digital Object Identifier 10.1109/71.877759  
[AbstractPlus](#) | [References](#) | [Full Text: PDF\(2036 KB\)](#) IEEE JNL  
[Rights and Permissions](#)
22. **Automatic Correction of Loop Transformations**  
Vasilache, Nicolas; Cohen, Albert; Pouchet, Louis-Noel;  
Parallel Architecture and Compilation Techniques, 2007. PACT 2007. 16th International Conference  
15-19 Sept. 2007 Page(s):292 - 304  
Digital Object Identifier 10.1109/PACT.2007.4336220  
[AbstractPlus](#) | [Full Text: PDF\(199 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
23. **Code generation for single-dimension software pipelining of multi-dimensional loops**  
Rong, H.; Douillet, A.; Govindarajan, R.; Gao, G.R.;  
Code Generation and Optimization, 2004. CGO 2004. International Symposium on  
2004 Page(s):175 - 186  
Digital Object Identifier 10.1109/CGO.2004.1281673  
[AbstractPlus](#) | [Full Text: PDF\(713 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
24. **Multi-dimensional incremental loop fusion for data locality**  
Verdoolaege, S.; Bruynooghe, M.; Janssens, G.; Catthoor, P.;  
Application-Specific Systems, Architectures, and Processors, 2003. Proceedings. IEEE International Conference on  
24-26 June 2003 Page(s):17 - 27  
[AbstractPlus](#) | [Full Text: PDF\(941 KB\)](#) IEEE CNF  
[Rights and Permissions](#)
25. **Loop Optimization using Hierarchical Compilation and Kernel Decomposition**  
Barthou, Denis; Donadio, Sebastien; Carribault, Patrick; Duchateau, Alexandre; Jalby, William;  
Code Generation and Optimization, 2007. CGO '07. International Symposium on  
11-14 March 2007 Page(s):170 - 184  
Digital Object Identifier 10.1109/CGO.2007.22  
[AbstractPlus](#) | [Full Text: PDF\(683 KB\)](#) IEEE CNF  
[Rights and Permissions](#)

View: 1-25 | 26-50 | 51-75

[Help](#) [Contact Us](#) [Privacy & Security](#) |

© Copyright 2007 IEEE – All Rights Reserved